НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»
ім. ІГОРЯ СІКОРСЬКОГО
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

До захисту допущено
Завідувач кафедри

_____ С.Г. Стіренко
(підпис) (ініціали, прізвище)

«_____» _____ 2019 року

Дипломний проект

освітньо-кваліфікаційного рівня «бакалавр»

з напрямку підготовки (спеціальності) 6.050102 «Комп'ютерна інженерія»
на тему: «Система розпізнавання голосу»

Виконала: студентка IV курсу, групи ІО-52

Стефанішина Уляна Савівна _____
(прізвище, ім'я, по батькові) (підпис)

Керівник д.т.н., проф. Сімоненко В.П. _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Консультант нормоконтроль д.т.н., проф. Сімоненко В.П. _____
(назва розділу) (вчений ступінь та звання, прізвище та ініціали) (підпис)

Рецензент _____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цьому дипломному проекті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019 року

Національний технічний університет України

«Київський політехнічний інститут»

ім. Ігоря Сікорського

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Освітньо-кваліфікаційного рівня **бакалавр**

Напрям підготовки **6.050102 «Комп'ютерна інженерія»**

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ С.Г. Стіренко
(підпис) (ініціали, прізвище)

«_____» _____ 2019 року

ЗАВДАННЯ

на бакалаврський дипломний проект студентки

Стефанішина Уляна Савівна
(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) *Система розпізнавання голосу*

Керівник проекту (роботи) д.т.н., проф. каф. ОТ, Сімоненко В.П.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

затверджені наказом по університету від «___» _____ 2019 року № _____

2. Термін здачі студенткою закінченого проекту (роботи) _____
3. Вихідні дані до проекту (роботи) див. технічне завдання
4. Зміст розрахунково-пояснювальної записки (перелік питань, які розробляються) *Огляд математичних та програмних підходів до розпізнавання голосового сигналу. Розробка програми медичного обслуговування пацієнтів з застосуванням голосового вводу даних.*
5. Перелік графічного матеріалу (з точним позначенням обов'язкових креслень)_____
   _____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
1-4	Сімоненко В.П.		

7. Дата видачі завдання_____

Календарний план

№, з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання		
2	Збір інформації		
3	Розробка варіантів реалізації розпізнавання голосу та підбір інструментів		
4	Реалізація структури бази даних		
5	Перевірка правильності взаємодії бази даних з серверною частиною		
6	Розробка інтерфейсу користувача та розпізнавання голосу на клієнтській частині		
7	Оформлення дипломної роботи		
8	Проходження нормоконтролю		
9	Отримання допуску до захисту та подача роботи в ДЕК		

Студент-дипломник _____
(підпис)

Керівник роботи _____
(підпис)

Анотація

В дипломному проекті розглядаються математичні методи розпізнавання голосових сигналів та програмні інструменти.

Метою розробки дипломного проекту є дослідження сучасних підходів та методів розпізнавання голосового сигналу та побудова системи розпізнавання голосового сигналу.

Результатом дипломного проекту є побудована автоматизована система розпізнавання голосового сигналу для працівників медичної галузі для спрощення проведення первинного огляду пацієнтів і, як наслідок, зменшення витрат часу фахівців на виконання рутинної механічної роботи.

В побудованій системі використаний сучасних підхід щодо розпізнавання мови та перетворення її в текст (Speech-to-text), використовуючи навчання нейромережі, задаючи наперед визначену граматику. Побудована SaaS-система являє собою клієнт-серверний застосунок з інтеграцією з базою даних.

Annotation

The presented diploma project covers mathematical methods for voice recognition and programming tools for implementing them.

The purpose of the development of the diploma project is the study of modern approaches and methods of voice recognition and the construction of a voice recognition system.

The result of the diploma project is the built automated voice recognition system for medical professionals to facilitate the initial examination of patients and, as a consequence, reduce the cost of time specialists to perform routine mechanical work.

In the built system used a modern approach to speech recognition and its transformation into text (Speech-to-text), using neural network learning by using a predefined grammar. Built-in SaaS-system is a client-server application with the database integration.

ВІДОМІСТЬ ДИПЛОМНОГО ПРОЕКТУ

№, з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
1	А4	ІАЛЦ 462619.002 ТЗ	*Система розпізнавання голосу* Технічне завдання	2	
2	А4	ІАЛЦ 462619.003 ПЗ	*Система розпізнавання голосу* Пояснювальна записка	60	
3	А4	ІАЛЦ 462619.004 Д1	*Система розпізнавання голосу* Діаграма класів	1	
4	А4	ІАЛЦ 462619.005 Д2	*Система розпізнавання голосу* Структурна діаграма	1	
5	А4	ІАЛЦ 462619.006 Д3	*Система розпізнавання голосу* Алгоритм розпізнавання голосового сигналу	1	

				ІАЛЦ 462619.001 ВР		
	ПІБ	Підпис	Дата			
Розробник	Стефанішина У.С.			*Система розпізнавання голосу* Відомість проекту	Лист	Листів
Керівник	Сімоненко В.П.				1	1
Консульт.	Сімоненко В.П.				НТУУ «КПІ» Каф. _ОТ_ Гр. ІО-52	
Зав. каф.	Стіренко С.Г.					

Технічне завдання

ЗМІСТ

1. НАЙМЕНУВАННЯ ТА ОБЛАСТЬ ЗАСТОСУВАННЯ

Найменування «Система розпізнавання голосу».

Область застосування – широкий спектр застосування в зв’язку з природних витоків керування технічним обладнанням для пришвидшення виконання роботи, зокрема, в проекті розроблена система для медичних установ для автоматизації первинного огляду пацієнтів та зменшення витрат робочого часу лікарів.

2. ПІДСТАВИ ДЛЯ РОЗРОБКИ

Підставою для розробки є завдання на виконання бакалаврського дипломного проекту, затверджене кафедрою обчислювальної техніки Національного технічного університету України «Київський політехнічний інститут» ім. Ігоря Сікорського.

3. МЕТА ТА ПРИЗНАЧЕННЯ РОЗРОБКИ

Мета розробки – автоматизація первинного огляду пацієнтів медичними працівниками та суттєве скорочення часу на виконання рутинної механічної роботи.

Розробка призначена для використання в медичних установах різного типу.

4. ДЖЕРЕЛА РОЗРОБКИ

Джерелом розробки є науково-технічна та математична література по розпізнаванню голосового сигналу, публікації та праці в Інтернеті.

5. ТЕХНІЧНІ ВИМОГИ

Вимогами до програмного продукту, що розробляється є його гнучкість щодо використання в різних середовищах, тобто розробка SaaS-системи. Простота використання та готовність системи щодо обробки помилок.

6. ЕТАПИ РОЗРОБКИ

Вивчення літературних джерел та публікацій, огляду сучасних підходів та вибір потрібного методу. 15.12.2018

Розробка алгоритму розпізнавання. 15.03.2019

Розробка клієнтської програми з застосуванням обраного алгоритму та його тестування. 05.04.2019

Розробка серверної частини програмного забезпечення, налаштування інтеграції з базою даних, побудова схеми бази даних. 12.04.2019

Завершення розробки клієнтської частини та її інтеграція з серверною частиною. 14.04.2019

Оформлення матеріалів дипломного проекту 20.05.2019

Захист бакалаврської роботи 18.06.2019

Пояснювальна записка

ВСТУП

В сучасному світі спостерігається стрімкий розвиток технологій. З моменту появи перших ЕОМ постала задача ефективної взаємодії людини з машиною. Протягом довгого часу взаємодія з коп'ютером була доступна тільки певним спеціаліста і всі інші могли взаємодіяти лише через такого посередника – програміста. Такий метод взаємодії проіснував аж до появи діалогового інтерфейсу, коли користувач вже міг за допомогою клавіатури ввести адресовану машині команду і отримати відповідь. Проте, саме поява графічного інтерфейсу, який позбавляв користувачів від обов'язку володіти командами для користування комп'ютером, стала рушієм для розповсюдження персональних комп'ютерів.

Проте, наука постійно рухалась в напрмку до більш універсального і природнього способу керування комп'ютером. Ще в часи, коли для передачі команд комп'ютеру використовувались перфокарти, в науково-фантастичних романах зображували людину, яка розмовляла з комп'ютером, наче з іншою людиною. Саме в ті часи й зародились ідеї та перші кроки щодо втілення такого способу керування комп'ютером в життя.

Проте, незважаючи на те, що виникнення задачі розпізнавання машиною людського голосу датується 50-ми роками минулого століття, ця задача і по сьогодні залишається актуальною.

РОЗДІЛ 1.
ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Історія розвитку машинного розпізнавання людського голосу

Перша спроба створення пристрою, який би реагував на людський голос була здійснена в 1916 році. Тоді була створена іграшка під назвою “Radio Rex”. Коли звучала команда «Rex!» собака вистрибувала з коробки. Технологія розпізнавання мови полягала в двох металевих пластинах, які були підібрані таким чином, аби при звукові частотою 500 Гц, що відповідає частоті голосної букви «е» в команді, яка подавалась розмикався ланцюг, електромагніт переставав тримати пружину і собака вистрибувала з коробки. [1]

Перші системи розпізнавання мови були спроектовані для розуміння тільки цифр. В 1952 році компанія Bell Laboratories розробили систему “Audrey”, яка розпізнавала цифри з ймовірністю 90%, які обов’язково повинні були відповідати таким критеріям:

- Диктор – особа чоловічої статі;
- Диктор вже мав досвід роботи з системою;
- Пауза між цифрами мала бути 350 мілісекунд.

Вже через 10 років в 1962 році IBM випустила покращений голосовий калькулятор під назвою “Shoebox” [2], який розпізнавав 16 англійських слів (цифри від нуля до дев’яти та прості математичні операції). Назву цей проддукт отримав завдяки своїм розмірам, які були такі ж, як коробка від взуття на той час. Він міг опрацьовувати прості фрифметичні команди, які подавались йому на вхід, а потім виводити результат на екран. Для розпізнавання ці слова було зручно розподілити на три частини – початковий, середній і кінцевий звуки. І, в залежності від сукупності цих звуків, визначалась команда, яка була подана. В цей самий час, над подібними цілями працювали різні лабораторії з США, Японії,

Англії та СРСР, які з часом розробили ще кілька апаратів, які розпізнавали вже окремі звуки, таким чином розширивши технологію розпізнавання мови з підтримкою до чотирьох голосних та дев'яти приголосних звуків.

Наступні 40 років системи розпізнавання розвивались поступово збільшуючи кількість слів та якість їх розпізнавання. В 1970-х роках системи вже могли розпізнати близько 1000 слів і використовувався алгоритм пошуку Beam Search. Проте вже в 1980-х роках, завдяки застосуванню прихованих марковських моделей, з'явилась можливість розпізнавати близько 5000 слів. Проте такі системи все ще могли розпізнавати тільки дискретну мову, що зобов'язувало робити паузи між словами.

До 1990-х років всі системи розпізнавання мови базувались на співставленні шаблонів, де звукові хвилі були переведені в набори чисел та зберігались таким чином. Після цього вони спрацьовували б коли в апараті прозвучить однаковий звук. З цього слідує, що для дієздатності тогочасних систем необхідна була чітка нешвидка вимова слів без сторонніх шумів з перервою між кожним словом.

Тільки в 1997 році з появою нових порівняно швидких процесорів побачив світ перший в світі «неперервний розпізнавач мови», що означало, що більше не потрібно буде робити перерви між словами. Цей розпізнавач був представлений в якості програмного забезпечення "Dragon's NaturallySpeaking".

На початку XXI століття, завдяки появі великих технологічних компаній, які володіли великими потужностями розподілених обчисленнь, розвиток розпізнавання мови набув темпів. Так в 2001 році компанія Microsoft випустила систему розпізнавання мови, яка працювала з Office XP, в 2002 році Google випускає Voice Search – технологія голосового пошуку в мережі, а в 2005 році була випущена перша операційна система з функцією розпізнавання та синтезу мови – Mac OS X Tiger (за розпізнавання мови відповідала програма VoiceOver). І вже, починаючи з 2009 року, програми, які використовували розпізнавання мови, почали з'являтись на мобільних пристроях.

1.2. Розпізнавання мови в сучасності

На сьогоднішній день розпізнавання мови відбувається з досить високою точністю. Відомою розвагою користувачів мобільних пристроїв є голосові додатки, такі як «Siri», «Cortrana», «Okay, Google», які дозволяють відправити повідомлення, знайти відповіді на запитання, налаштувати будильник та використовувати інші функції мобільного телефону голосовим управлінням. Крім того, з «Siri» і «Cortrana» можна просто поспілкуватися, поставивши їм питання особистого характеру, на які вони дадуть відповідь.

Один з найпростіших прикладів - це перетворення мови в тектовий вигляд. Серед таких застосунків знаходиться «Яндекс.Диктовка». Він записує текст, який був продиктований. Також у ньому доступні стандартні команди текстових редакторів: виділити слово/речення/текст, скопіювати, вставити, видалити, які також активізуються голосом [3]. Для успішної роботи необхідне відмінне інтернет-з'єднання, а мова має бути достатньо розбірливою, голосною та повільною.

Розпізнавання мови також має широке застосування в call-центрах. Робот може провести початкове опитування, щоб виявити проблему і розподілити звінок на потрібного оператора. А для компаній, де діалоги легко шаблонізувати, він може виступити і замість оператора [4]. Крім того, за допомогою цих технологій можна ознайомитися з контролем якості роботи працівників, аналізуючи їх мову. Після аналізу оператору дають оцінку і поради щодо поліпшення стану діалогів [5]. Також, аналізуючи діалоги, можно проводити статистику запитів клієнтів.

Ще одне важливе застосування - це створення додатків для людей з обмеженими можливостями (глухих для прикладу). Одним з таких застосунків є «Яндекс.Разговор». Він полегшує спілкування таких людей з навколишніми, так як їх співрозмовникам не порібно записувати свою мову. «Яндекс.Разговор» розпізнає і записує слова, які дозволяють користувачу дізнатися, що ж йому

відповіли. Надруковану відповідь можна озвучити з допомогою синтезу мови, або ж також можна показати в спеціальному великому форматі [6].

Одним з найбільш широких напрямків - це розпізнавання голосових команд. Так, наприклад, можна попросити розумний будинок включити вентилятор, ввімкнути світло, розповісти прогноз погоди або ж ще подібні побутові речі.

1.3. Системи розпізнавання мови

Для прикладу, Yandex Speech Kit - це комплекс технологій розпізнавання мови компанії Яндекс, який включає в себе розпізнавання та синтез мови, голосову активацію та виділення смислових об’єктів в мові. [7] Він доступний користувачу в декількох варіантах.

JavaScript API SpeechKit застосовується в веб-додатках. Він дозволяє розмістити на сайті спеціальне поле для голосового вводу, текст якого може бути використаний для створення коментарів, навігацій по сайту або для здійснення пошуку. Також з допомогою цієї бібліотеки можна озвучити інформацію.

Mobile SDK SpeechKit - API, призначений для використання в роботі з мобільними додатками під різними платформами. У кожній з технологій доступне розпізнавання мови українською, російською та турецькою мовах.

Технологія розпізнавання мови є доволі популярною та перспективною, тому існує безліч системних реалізацій.

CMU Sphinx - один з найпопулярніших проектів для розпізнавання мови з відкритим кодом. Являється дикторонезалежним, і може розпізнавати мову в непрервному потоці. Також він допомагає дослідникам і розробникам створювати системне розпізнавання мови [8].

Іншим популярним розпізнавачем мови з відкритим кодом є Julius. Він володіє великим словником, але при цьому високою швидкістю роботи, що

дозволяє розпізнати мову в режимі близькому до реального часу. Спочатку він був спроектований як інструмент для дослідження в області розпізнавання японської мови, тому існують моделі тільки для японської та англійської мови [9].

Google Cloud Speech API дозволяє перетворювати мову в текст для більш ніж 80 мов. Розпізнавання відбувається віддалено на серверах Google, тому для працездатності розпізнавача необхіжне підключення до мережі Інтернет. При наявності відмінного з'єднання розпізнавання відбувається практично миттєво. Також API може відділяти мову від шуму, що потрапляє в аудіопотік, що дозволяє збільшити якість розпізнавання. Google Cloud Speech API може використовуватись в різних додатках і пристроях, які можуть відправляти REST або gRPC-запити [10].

Microsoft Speech API виконує розбір і синтез речей в додатках Windows. Він спроектований таким чином, щоб вбудовуватись за допомогою інтерфейсів як можна простіше. Також є можливість створити свої або змінити існуючі механізми розпізнавання і синтезу мови.

1.4. Перспективи систем розпізнавання мови

Системи розпізнавання мови вже пройшли довгий шлях за свою невелику історію. Але що ж чекає цю сферу в майбутньому?

Насамперед, компанії, які працюють в цій сфері, продовжують підвищувати якість розпізнавання мови. Тільки таким чином голосові технології мають можливість набути популярності та примножити велику кількість користувачів. Це дозволить змінити спосіб використання техніки: замість натискання кнопок, буде можливість спілкуватись з технікою.

Схожі зміни чекають на автомобільну індустрію.Якщо зараз найбільшою популярністю користується тільки голосовий ввід в навігатор, то в подальшому

водій повністю буде звільнений від будь-яких побічних механічних дій і повністю буде зосереджений на дорозі. Всі команди будуть вводитись голосом, а відповіді на них будуть проголошені вслух. За допогою такої системи можна буде, наприклад, написати повідомлення і отримати голосову відповідь на його, дізнатися прогноз погоди, розклад подій, включити музику або вибрати радіостанцію.

Іншим напрямком розвитку є голосова біометрія. На сьогоднішній день вона розвивається в повільному темпі, проте має широкі перспективи. З точки зору дослідження мовленнєвих технологій її завдання полягаєв тому, щоб вияснити які числа в цифровому представленні голосового сигналу відповідали тій чи іншій характеристиці мови. З практичної точки зору, такі системи повинні розпізнавати стать людини, настрій та віковий діапазон людини, що розмовляє. Це дозволить формулювати більш точні та структурованіші відповіді на покладені системі питання. Також до біометрії відноситься розпізнавання людини по її голосу. На сьогоднішній день, така система вкрай ненадійна для систем аутентифікації користувачів, але вже має місце в застосуванні в якості додаткового захисту в банківських системах на одній вазі з кодовим словом та секретною інформацією.

Мовленнєві технології будуть вкрай корисні в медицині. Лікарям доводиться виконувати багато роботи з паперовими матеріалами, вести звітність та документацію. На це все йде багато часу, який міг би бути витрячений з користю на прийом пацієнтів. Набагато простіше вести звітність, надиктовуючи її, а система вже все переведе в текст. Звуження тематики мовлення від загальної до суто медичної дозволятиме значно зменшити дублювання інформації. Зараз лікар, після проведення діагностики та пояснення діагнозу пацієнту записує все те ж саме в картку. З використанням технологій розпізнавання голосу, достатньо буде всього лише промовити все один раз і система автоматично розпізнає діагноз.

Основною проблемою, яка існує по сьогоднішній день – це виявлення основного голосового сигналу серед супутніх шумів, а також відділення різних

голосів в одному сигналі. Технологія повинна розпізнавати тільки ту людину, яка була вказана і повинна відділяти команди цієї людини від інших. Саме вирішення цієї проблеми дозволить більшій кількості різноманітних систем впроваджувати розпізнавання та функціонувати в звичайному для людей шумному середовищі.

1.5. Загрози систем розпізнавання мови

Голосовий інтерфейс – це не лише новий спосіб управління пристроями, а також ще нові способи для злому.

Один з основних недоліків – це хибне спрацьовування. Для прикладу – якщо в приміщенні, де знаходиться багато людей, хтось намагається виконати голосову команду на своєму пристрої, то, скоріш за все, пристрої ще кількох людей відреагують на запит.

Одним з очевидних способів вирішенням даної проблеми – це «навчити» пристій голосу тільки його власника, але такий підхід вимагає навчання кожного пристрою, що є неефективним і не вільнодоступним всім користувачам.

Інший спосіб – це погіршувати якість команди, що подається. Якщо це вірусна команда, то система її просто не зможе розпізнати, але на команди звичайних людей такий підхід не поширюватиметься. Проте, такий підхід суттєво погіршує роботу з пристроями людям з жефектами мови і робить для них голосове керування майже непридатним.

Найефективніший спосіб – це впровадження технології, яка б перед обробкою керуючої команди, аналізувала голос і чи дозволено цьому голосу віддавати керуючі команди.

Висновки до розділу 1

Технології розпізнавання мови за останні роки дуже прогресують: якщо спочатку в словнику системи знаходились лише цифри – то сьогодні вона подібна людині, та має широкий мовленнєвий запас. Такі системи розпізнавання мови вже отримали популярність, будучи вбудованими в мобільні пристрої та комп'ютери. Також такі системи розпізнавання мови є корисними в різноманітних call-центрах та банківських системах, виконуючи обслуговування клієнтів простішим та швидшим. Вже на сьогоднішній день ці системи допомагають автомобілістам та допомагають людям з відхиленнями слуху простіше спілкуватися з людьми.
Проте, на цьому їх стрімкий розвиток не зупиняється і в майбутньому на них чекає багато нових застосувань в нових сферах діяльності. Проте, не варто забувати і про загрозу, яка виростає поруч з розвитком новим технологій а даній сфері.

На хвилі популярності технологій розпізнавання мови з'явився цілий спектр систем, як з відкритим кодом, так і ні, але надаючих API для їх використання. Одним з найпопулярніших систем є Google Cloud Speech API. До переваг цієї системи належить велика кількість мов, які вона підтримує для розпізнавання, приглушення шуму, постійне навчання системи, висока якість розпізнавання та велика кількість прикладів. Проте, Google надає безкоштовний доступ до невеликої кількості функціоналу.

РОЗДІЛ 2.
ЕЛЕМЕНТИ ТЕОРІЇ РОЗПІЗНАВАННЯ МОВИ

2.1. Приховані Марковські моделі

2.1.1. Ланцюг Маркова

В термінах систем розпізнавання мови випадковим процесом являється, так званий, марковський процес, якщо для окремо взятого моменту часу t_0 ймовірність певного конкретного стану системи з часом (при $t > t_0$) залежить лише від її стану в початковому часі (при $t = t_0$) і не залежить від того, яким чином і в який момент часу система прийшла в такий стан (тобто немає жодного впливу розвиток випадкового процесу при $t < t_0$).

Марковські випадкові процеси поділяються на два види: дискретні та неперервні.

Дискретний марковський процес це такий випадковий процес в якого всі допустимі стани системи, наприклад, $S_1, S_2, \ldots, S_i, \ldots, S_n$ можна явно визначити, а процес характеризується тим, що в певні проміжки часу система S миттєво (стрибково), переходить з певного стану в інший.

Неперервними випадковими процесами являються такі системи, які характеризуються послідовними переходами з одних станів в інші. Для прикладу таких систем можна вказати: неперервна зміна електричних показників (наприклад, сила струму, напруга та ємність) ланцюга, перебіг зміни кількості паливного матеріалу в транспортних засобах та інші.

Дискретні випадкові процеси прийнято описувати за допомогою графічної схеми, на якій в вигляді квадратів (вузлів) показують наявні дискретні стани системи, а за допомогою одно направлених стрілок – всі переходи випадкової

системи з одного стану до іншого. Таке зображення випадкового дискретного процесу прийнято називати – графом станів системи. Для прикладу, на рис. 2.2 зображено граф, на якому показано систему, яка може мати всього $n = 6$ станів, $l = 10$ переходів; в більшості випадків $n \neq l$.

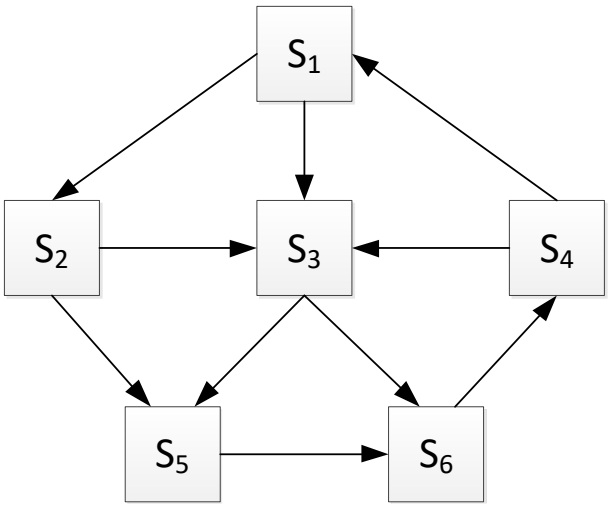


Рисунок 2.1 Граф станів системи з $n = 6, l = 10$

Для проведення математичного аналізу випадкового марковського процесу, який відслідковується в дискретній системі, необхідно визначити моменти часу можливих переходів системи з одного стану в інші. Дані про можливі переходи можуть бути відомі завчасно або відбуватись абсолютно випадково.

Випадковимі процеси з дискретним часом характеризуються тим, що переходи системи з одного стану в інший можуть відбуватися тільки в строго визначені моменти часу. Між вказаними моментами часу система *S* залишається в незмінному стані.

Випадкові процеси з неперервним часом характеризуються тим, що переведення системи з одного стану в інший може відбутися в певний випадковий момент часу t.

Марковськими випадковими процесами з дискретним часом, які відслідковуються в дискретних системах, мають назву - ланцюи Маркова, а

марковські випадкові процеси неперервні в часі, які відслідковуються в дискретних системах, мають назву неперервних ланцюів Маркова. [10]

2.1.2. Приховані марковські моделі в розпізнаванні мови

Прихована марковська модель (ПММ) — це, так звана, статистична модель, яка імітує роботу процесу, який схожтй на марковський з невідомими параметрами і його завданням є пошук невідомих за допомогою спостереження. ПММ розглядається як проста Байєсівська сітка довіри [11].

Статистичні методи, які базуються на визначенні марківського джерела, а також приховані марковські моделі (ПММ) вперше винайдені та дослідженні ще наприкінці 60-х — та на початку 70-х років. Перше їх практичне застосування для обробки мови Бейкером (Baker) з CMU та Джелінеком (Jelinek) і колегами з IBM датується 70-ми роками минулого століття. Такі моделі доволі змістовні за своєму математичною структурою та в подальшому мають правр будувати теоретичну основу для багатьох додатків в сфері розпізнавання мови. Як наслідок, достовірне їх застосування призводить до рішення певних важливих задач в прикладних сферах розвитку науки.

Однак, широкого розповсюдження ПММ в завданнях розпізнавання голосу отримали не так давно: спочатку основна теорія по прихованих марковських моделях була опублікована в спеціальних наукових журналах для математичних фахівців. Такі журнали були не надто популярні посеред інженерів, які займаються розпізнаванням голосу. Окрім цього, опубліковані в журналах теоретичні основи не містили в собі пояснень щодо можливостей та способів застосування ПММ в різноманітних прикладних областях. На сьогодні такі марковські моделі користуються широкою популярністю [12, 13].

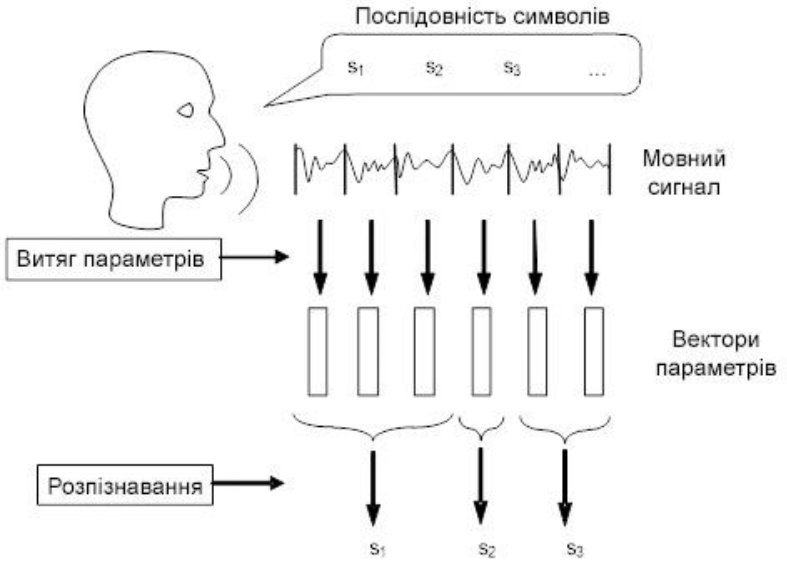


Рисунок 2.1. Кодування/декодування повідомлення

Загалом, при розгляданні можливих способів розпізнавання голосу мають на увазі, що мовний сигнал — це таке певне повідомлення, яке закодоване за допомогою певної кількості набору чи окремих символів (наприклад, послідовність звуків — рис. 2.3). Для проведення оберненої процедури щодо розпізнавання певних послідовностей символів, які наявні в певному фрагменті мовного сигналу, неперервний мовний сигнал спершу перетворюють в послідовність рівновіддалених векторів дискретних параметрів, які представляють собою часові вектори. Мається на увазі, що така віднайдена послідовність часових векторів, які представляють собою параметри формує достатньо достовірне представлення форми хвилі мовного сигналу, так як на проміжку часу, який охоплений одним з векторів (найчастіше, беруть 10 мс), мовний сигнал може в наближенні вважатись стаціонарним. Найчастіше, використовують представлення параметрів, які грунтуються на згладжених спектрах або ж на коефіцієнтах лінійного наближення та інші способи представлення, для прикладу, мел-кепстральні коефіцієнти.

Суть системи розпізнавання голосу полягає в тому, що така система

повинна співвіднести послідовність векторів параметрів мови та необхідні набори символів. Досягти такого ефекту важко. По-перше, перехід від символів до мови не є однозначним: тобто, різні символи можуть спричиняти появу однакових звуків, тобто схожих частот. А також, доволі суттєві зміни мовленнєвого сигналу відбуваються при зміні диктора, інтонації чи навколишнього середовища та інші подібні природні фактори. По-друге, важко достатньо точно визначити границі між символами на мовному сигналі. Через вказані прични мовний сигнал неприпустимо розглядати як набір поєднаних незмінних проміжків.

Іншої перешкоди, яка зумовлена невизначенням конкретних границь мовної частини, можна уникнути, якщо обмежитись окремою задачею розпізнавання певних слів. Це означає, що мовний сигнал відповідатиме одному символу у вигляді слова, яке буде обрано з певного словника. Таке дещо спрощене завдання дещо штучним, та воно широко застововується зараз на практиці. Більше того, воно є доволі доброю базою для вивчення основних ідей розпізнавання голосу при допомозі ПММ до того як підїодити до складнішого випадку неперервного мовленнєвого сигналу.

Для початку необхідно дати фомальне визначення елементів ПММ та пояснення того, як модель буде генерувати досліджувану послідовність символів. ПММ характерихується наступними поняттями:

1. N — можлива кількість *станів* в моделі. Не зажаючи на те, що стани в ПММ є прихованими, в більшості випадків залишається невідповідність між описаним станом такої моделі та реальним станом процесу. Загалом, перехід в певний обраний стан системи можливий з певнго іншого стану загалом системи (а також переход стану самого в себе); з іншого боку, лише певні можливості переходів мають сенс в певній вказаній моделі. Нехай сукупність станів моделі буде визначена множиною $S = \{S_1, S_2, \ldots, S_N\}$, а поточний стан як q_t в певний час t

2. M, загальна кількість допустимих *символів* в досліджуваному наборі, розмір *алфавіту досліджуваного набору*. *Алфавіт досліджуваного набору* позначається $W = \{w_1, w_2, \ldots, w_M\}$.

3. *Матриця ймовірностей переходів* (або матриця переходів) $A = \{a_{ij}\}$, де

$$a_{ij} = P\left[q_{t+1} = S_j \mid q_t = S_i\right],\ 1 \le i,\ j \le N \tag{2.1}$$

тобто це буде ймовірність того, що система, яка наразі перебуває в стані S_i, перейде в стан S_j. Якщо для певних двох станів в такій моделі доступний перехід з пекного стану в інший, то $a_{ij} > 0$ для довільних i, j. В будь-яких інших ПММ для певних i, j ймовірність переходу $a_{ij} = 0$.

4. $B = \{b_j(k)\}$ — розподіл ймовірностей перебування символів в *j*-тому стані, де

$$b_j(k) = P\left[w_k \mid q_t = S_j\right],\ 1 \ge j \ge N,\ 1 \ge k \ge M. \tag{2.2}$$

$b_j(k)$— ймовірність того, що в певний час t, система, яка перебуває в *j*-тому стані (стан S_j), розпізнає *k*-тий символ (символ w_k) в досіджуваний набір.

5. Розподіл ймовірностей початкового стану $\pi = \{\pi_i\}$, де

$$\pi_i = P\left[q_1 = S_i\right],\ 1 \le i \le N \tag{2.3}$$

тобто ймовірність того, що S_i — це початковий стан моделі.

Сукупність значень N, M, A, B і π — це прихована марковська модель, яка може згенерувати *досліджуваний набір*.

$$O = o_1, o_2, \ldots, \mathrm{o}_T \tag{2.4}$$

(де o_t — певний символ алфавіту W, T — кількість елементів в досліджуваному наборі).

ПММ будує досліджувану послідовність за наступним алгоритмом:

1. Обирається початковий стан $q_1 = S_i$ відповідно до розподілу π.

2. Встановлюється $t = 1$.

3. Обирається $o_t = w_k$ відповідно до розподілу $b_j(k)$ в стані (S_i).

4. Модель переводиться в інший стан $q_{t+1} = S_j$ відповідно до *матриці переходів* a_{ij} з врахуванням наявного стану S_i.

Встановлюється час $t = t + 1$; повертаємось до кроку 3, якщо $t < T$; в протилежному випадку — закінчується виконання [13].

В підсумку, варто зазначити, що *загальнйи* опис ПММ складається з двох параметрів моделі (N и M), опису символів досліджуваної послідовності та трьох масивів ймовірностей — A, B і π. Тому, необхідне введенн запису для позначення *достатнього* опису параметрів моделі:

$$\lambda = (A, B, \pi). \tag{2.5}$$

2.1.3. Розпізнавання окремих слів

Визначимо вимовлене слово, як послідовність векторів або *спостережень* O які визначаються як

$$O = o_1, o_2, o..., o_T, \tag{2.6}$$

де o_t являється вектором параметрів мови, досліджуваний в поточний момент часу *t*. Перешкоду розпізнавання окремих слів можна розглядати як результат обчислення

$$\arg\max_i \left\{ P(\omega_i | O) \right\}, \tag{2.7}$$

де ω_i є *i*-те слово словника. Умовну ймовірність $P(\omega_i | O)$ не обчислюють безпосередньо, а користуються формулою Байеса:

$$P(\omega_i | O) = \frac{P(O | \omega_i) P(\omega_i)}{P(O)}. \tag{2.8}$$

В підсумку, при визначеній апріорній ймовірності $P(\omega_i)$, найймовірніше вимовлене слово визначається такою ймовірністю $P(O|\omega_i)$. Через високу визначеність послідовності спостережень O пряме оцінювання спільної умовної ймовірності $P(o_1, o_2,...|\omega_i)$ на екземплярах вимовлених слів не використовується на практиці. Але якщо припустити *параметричну модель генерації слова, як Марковська модель*, визначення за даними стає доступним, так як *проблема оцінки умовної густини $P(O|\omega_i)$ замінюється простішою проблемою оцінки параметрів Марковської моделі*.

При підході до розпізнавання мови, який грунтується на ПММ, *допускається, що послідовність досліджуваних векторів мови, які відповідають певному слову, породжена Марковською моделлю*, як показано на рис. 2.4. [12] [14]

Видно, що послідовність станів показаної моделі має особливість, яка характеризується тим, що із збільшенням часу індекс стану теж збільшується (або ж залишається незмінним), тобто, стани переходять з одного в інший зліва направо. Це є ліво-права модель, або ж модель Бакіса. Застосування такої моделі дозволяє зменшення кількості можливих наборів станів моделі, яка розглядається.

На рис. 2.4 вказаний приклад такого процесу, де модель, яка має шість станів, проходить через послідовність станів X=1,2,2,3,4,4,5,6, щоб згенерувати послідовність від o_1 до o_6. Зрозуміло, що вхідний і вихідний стани ПММ не є породжуваними. Це застосовано для спрощення моделюванян важких систем.

Сукупна ймовірність того, що O згенерована моделлю M, яка проходить через послідовність станів X, визначається як добуток ймовірностей переходу і ймовірностей генерації. Так, для показаної на рис. 2.4 послідовності станів X

$$P(O, X|M) = a_{12}b_2(o_1)a_{22}b_2(o_2)a_{23}b_3(o_3)\ ... \qquad (2.9)$$

На практиці, відомою є лише послідовність спостереження O, в той же час як породжуюча послідовність станів X є прихованою від спостереження. Саме

тому таку модель називають Прихованою Марковською Моделлю.

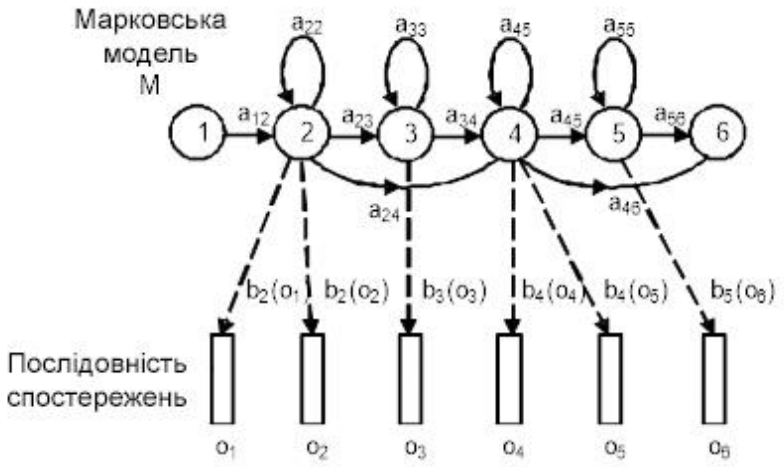


Рисунок 2.2. Марковська модель генерування послідовності випадкових векторів

Так як послідовність X невідома, необхідна достовірність визначається сумою всіх відомих послідовностей станів системи, тобто:

$$P\left(O\middle|M\right)=\sum_{X}a_{x(0)x(1)}\prod_{t=1}^{T}b_{x(t)}\left(o_t\right)a_{x(t)x(t+1)}, \tag{2.10}$$

де на $x(0)$ накладається вимога, аби модель була застосовна до вхідного стану, а до $x(T+1)$ - була моделлю вихідного стану.

В якості альтернативи до виразу (1.10), достовірність обчислюється наближено, шляхом визначення найбільш ймовірної послідовності станів, тобто:

$$P\left(O\middle|M\right)=\max_{X}\left\{a_{x(0)x(1)}\prod_{t=1}^{T}b_{x(t)}\left(o_t\right)a_{x(t)x(t+1)}\right\}. \tag{2.11}$$

Здійснити прямі обчислення, у відповідності з співвідношеннями (2.10) і (2.11), не надто просто, однак мають місце рекурсивні процедури, які дозволяють доволі ефективно розраховувати обидві величини. Підкреслимо, що, якщо співвідношення (2.7) визначається, тоді перешкода щодо розпізнавання мови вирішена. Для заданого набору моделей, які відповідають словам, співвідношення

(2.7) визначається з використанням (2.8) з врахуванням, що

$$P\left(O\middle|\omega_i\right)=P\left(O\middle|M_i\right). \tag{2.12}$$

Та, припускається, що такі параметри, як a_{ij} і $b_j(o_t)$ уже відомі для кожної окремо взятої моделі M_i. В цьому і заключається вишуканість та потужність ПММ моделі. Для вказаної множини прикладів навчання, відповідних конкретній моделі, параметри такої моделі визначаються автоматично за допомогою надійної і ефективної рекурентної процедури. Тобто, за умови, що зібрана достатня кількість потрібних зразків кожного слова, може бути побудована ПММ, яка *приховано моделює всю множину причин мінливості*, властивій реальній мові.

2.1.4. Опис ймовірності породження

Перед детальним поясненням проблеми параметричного оцінювання, необхідно визначити вид роподілів $b_j(o_t)$. Залежно від моделюючих параметрів розподіли багатовимірних густин ймовірності можуть бути неперервними та дискретними. Для початку, буде розглядатись неперервні щільності розподілів.

В більшості систем, які працюють з неперервними густинами, розподіли описуються гаусівськими сумами густин ймовірностей. В такому разі формула розрахунку $b_j(o_t)$ набуває вигляду:

$$b_j(o_t)=\sum_{m=1}^{M}c_{jm}N\left(o_t,\mu_{jm},\Sigma_{jm}\right), \tag{2.13}$$

де M – число суми компонентів, c_{jm} – вага m-го компоненту, а $N(o, \mu, \Sigma)$ – багатомірний Гаусівський розподіл з вектором середнього значення μ і коваріаційною матрицею Σ:

$$N\left(o,\mu,\Sigma\right)=\frac{1}{\sqrt{\left(2\pi\right)^n\left|\Sigma\right|}}\exp\left(-\frac{1}{2}\left(o-\mu\right)'\Sigma^{-1}\left(o-\mu\right)\right) \tag{2.14}$$

де n – розмірність o.

2.1.5. Рекурентне оцінювання Баума-Уелча (Baum-Welch)

Щоб віднайти параметри ПММ, для початку варто зробити припущення, якими могли б бути такі параметри. Після цього, можна знайти точніші параметри (в плані максимальної достовірності) за допомогою так званої рекурентної процедури Баума-Уелча.

Припустимо, що компоненти є спеціальною формою стану більш низького рівня, в якому ймовірності переходу є вагами (див. рис. 5).

Тобто, важливою задачею є оціннка середніх та дисперсій ПММ, в якій кожен стан вихідних даних являється єдиним Гаусівським компонентом:

$$b_j(o_t) = \frac{1}{\sqrt{(2\pi)^n |\Sigma_j|}} \exp\left(-\frac{1}{2}(o_t - \mu_j)' \Sigma_j^{-1} (o_t - \mu_j)\right) \qquad (2.15)$$

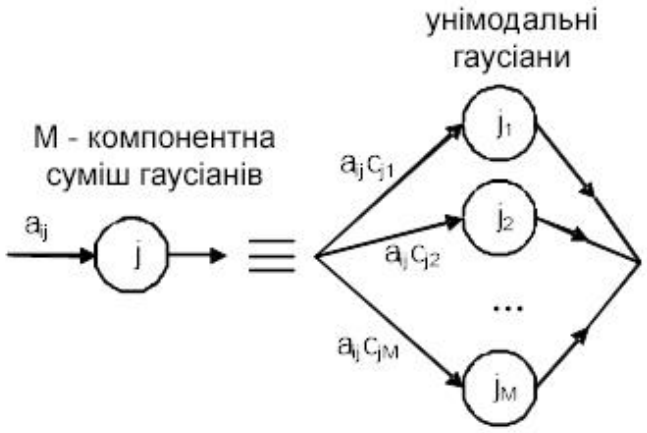


Рисунок 2.3. Представлення суміші

Якби ж в ПММ був лише єдиний стан, оцінити параметр було б легко. Оцінки максимальної достовірності величин μ_j і Σ_j можна було б отримати простим усередненням:

$$\hat{\mu}_j = \frac{1}{T}\sum_{t=1}^{T} o_t; \qquad (2.16)$$

$$\hat{\Sigma}_j = \frac{1}{T}\sum_{t=1}^{T}\left(o_t - \mu_j\right)(o_t - \mu_j)'. \tag{2.17}$$

На практиці, зрозуміло, існує більше одного станів, і неможливо прямо прив'язати вектор досліджень до конкретних станів, так як базові послідовності станів невідомі. Помітимо, що, якби можливо було провести певну наближену прив'язку векторів до станів, то можна було б використовувати рівняння (2.16) і (2.17) для отримання необхідних початкових значень параметрів. Далі, з використанням описаного нижче алгоритму Вітербі, відшуковується найбільш достовірна послідовність станів, вектори спостережень знову прив'язуються до станів, після чого знову використовуються рівняння (2.16) і (2.17) для отримання точніших початкових значень. Такий процес повторюється до тих пір, поки оцінки не перестають змінюватись. Так як повна достовірність кожної послідовності досліджень базується на додаванні всіх можливих послідовностей станів, кожен вектор дослідження додає свій вклад в розрахунки значень параметрів максимальної достовірності до кожного стану. Тобто, замість того, щоб прив'язувати кожний вектор спостереження до певного стану, як це було у вищезазначеному приближенні, кожне спостереження прив'язується до кожного стану, пропорційно ймовірності стану моделі при спостереженні цього вектору. Таким чином, якщо позначити через $L_j(t)$ ймовірність перебування в стані j в момент часу t, приведені вище рівняння (2.16) і (2.17) стають наступними зваженими середніми:

$$\hat{\mu}_j = \frac{\sum_{t=1}^{T} L_j(t) o_t}{\sum_{t=1}^{T} L_j(t)}; \tag{2.18}$$

$$\hat{\Sigma}_j = \frac{\sum_{t=1}^{T} L_j(t)\left(o_t - \mu_j\right)\left(o_t - \mu_j\right)'}{\sum_{t=1}^{T} L_j(t)}. \tag{2.19}$$

де підсумовування в знаменниках забезпечує необхідну нормалізацію. Рівняння **Ошибка! Источник ссылки не найден.** і

Ошибка! Источник ссылки не найден. описують процедуру рекурентного оцінювання Баума-Уелча для середніх і дисперсій ПММ. Аналогічна, але дещо важча, процедура отримана для ймовірностей переходу.

Але, щоб застосувати співвідношення (2.18) і (2.19) необхідно розрахувати ймовірність стану $L_j(t)$. Це швидко та легко робиться за допомогою *алгоритму прямого-оберненого ходу (Forward-Backward algorithm)*. Нехай пряма ймовірність $\alpha_j(t)$ для деякої моделі M з N станами визначена у вигляді:

$$\alpha_j(t) = P(o_1,...,o_t, x(t) = j). \tag{2.20}$$

Тобто, $\alpha_j(t)$ - сукупна ймовірність дослідження перших t векторів мови для стану j в момент часу t. Ця пряма ймовірність може бути ефективно розрахована за наступною рекурентною формулою:

$$\alpha_j(t) = \left[\sum_{i=2}^{N-1} \alpha_i(t-1)\alpha_{ij}\right] b_j(o_t). \tag{2.21}$$

Вигляд такої рекурентної формули визначається тією обставиною, що ймовірність перебування в стані j в момент t, при спостереженнях o_t, можна вивести шляхом додавання прямих ймовірностей для всіх ймовірних попередніх станів елементу i, зважених ймовірностей переходів a_{ij}. Дещо незвичайні границі зумовлені тим, що стани 1 і N не є породжуючими. Початкові умови для вищезазначеного рекурентного співвідношення мають вигляд:

$$\alpha_1(1) = 1; \tag{2.22}$$

$$\alpha_j(1) = a_{1j} b_j(o_1). \tag{2.23}$$

для $1<j<N$, а кінцеві умови задаються так:

$$\alpha_N(T) = \sum_{i=2}^{N-1} \alpha_i(T) a_{iN}. \tag{2.24}$$

Відмітимо, що з визначення $\alpha_j(t)$ слідує:

$$P(O \mid M) = \alpha_N(T). \tag{2.25}$$

Тобто, обчислення прямої ймовірності дозволяє отримати повну достовірність $P(O|M)$. Обернена ймовірність $\beta_i(t)$ визначається наступним чином:

$$\beta_j(t) = P\left(o_{t+1},...,o_T \mid x(t) = j, M\right). \quad (2.26)$$

Як і у випадку прямої ймовірності, ця обернена ймовірність може бути швидко визначена з використанням наступного рекурентного співвідношення:

$$\beta_i(t) = \sum_{j=2}^{N-1} a_{ij} b_j\left(o_{t+1}\right)\beta_j(t+1), \quad (2.27)$$

з початковою умовою:

$$\beta_i(T) = a_{iN}, \quad (2.28)$$

для $1<i<N$, і кінцевою умовою:

$$\beta_1(1) = \sum_{j=2}^{N-1} a_{1j} b_j\left(o_1\right)\beta_j(1). \quad (2.29)$$

Підкреслимо, що в приведених визначеннях пряма ймовірність є спільна ймовірність, в той же час як обернена ймовірність є умовною ймовірністю. Це дещо асиметричне визначення є навмисним, оскільки воно дозволяє визначити ймовірність надходження в стані як добуток цих двох ймовірностей. За визначенням,

$$\alpha_j(t)\beta_j(t) = P\left(O, x(t) = j \mid M\right), \quad (2.30)$$

звідси

$$L(t) = P\left(x(t) = j \mid M\right) = \frac{P\left(O, x(t) = j \mid M\right)}{P(O \mid M)} = \frac{1}{P}\alpha_j(t)\beta_j(t), \quad (2.31)$$

де $P = P(O \mid M)$.

Тепер відома вся інформація, необхідна для здійснення рекурентного оцінювання параметрів ПММ з використанням алгоритму Баума-Уелча.

Етапи алгоритму можна відобразити так:

1. Для кожного рекурентно оцінюваного векторного/матричного параметру,

варто виділити місце в пам'яті для організації додавання в чисельнику і знаменнику виразів **Ошибка! Источник ссылки не найден.** і **Ошибка! Источник ссылки не найден.**. Ці місця пам'яті є накопичувальними суматорами.

2. Обчислюються прямі та зворотні ймовірності для всіх станів j і моментів часу t.
3. Для конкретного стану j та часу t перезаписуються вміст накопичувальних суматорів, використовуючи ймовірність $L_j(t)$ та поточний вектор спостереження o_t.
4. Кінцеві значення накопичувального суматора використовують для вирахування нових значень параметрів.
5. Якщо значення $P = P(O|M)$ для даної ітерації не вище цього для попередньої ітерації, тоді закінчується обрахунок, в протилежному випадку варто повторити попередні кроки, використовуючи нові рекурентно оцінені значення параметрів.

З цього слідує, що параметри ПММ рекурентно оцінюються за єдиною послідовністю спостереження, тобто за допомогою єдиного екземпляра промовленого слова. Практично, для отримання високих оцінок параметра, необхідно багато екземплярів одного і того ж слова. Але, використання багаторазових послідовностей дослідження не доводить до ускладнення алгоритму: наведені кроки 2 і 3 повторюються для кожної нової навчальної послідовності.

2.1.6. Розпізнавання і декодування Вітербі

В попередньому підрозділі описані основні ідеї рекурентної оцінки параметрів ПММ з використанням алгоритму Баума-Уелча. При цьому зазначено, що високоефективний рекурсивний алгоритм обчислення прямої ймовірності дозволяє водночас обрахувати також повну ймовірність $P(O|M)$. Тобто, такий алгоритм може бути використаний для знаходження моделі, яка максимізує значення $P(O|M_i)$ і, як наслідок, може бути використана для розпізнавання.

Практично, зручніше проводити розпізнавання, базуючись на максимізації достовірності послідовності стану, оскільки це легко узагальнити у випадку неперервної мови, що є неможливим при використанні повної ймовірності. Цю достовірність обчислюють, використовуючи той самий алгоритм що і при обрахуванні прямої ймовірності, з різницею, що сумування замінюється пошуком максимуму. Допустимо, що $\varphi_j(t)$, для даної моделі M, представляє максимальну достовірність дослідження послідовності векторів мови від o_1 до o_t і перебування в стані j в певний час t. Цю часткову достовірність можна ефективно визначити з використанням вказаного нижче рекурентного співвідношення:

$$\phi_j(t) = \max_i \left\{ \phi_i(t-1) a_{ij} \right\} b_j(o_t), \tag{2.32}$$

де

$$\phi_1(1) = 1; \tag{2.33}$$

$$\phi_j(1) = a_{1j} b_j(o_1), \tag{2.34}$$

для $1 < j < N$. Тоді максимальна правдоподібність $P(O|M)$ має вигляд:

$$\phi_N(T) = \max_i \left\{ \phi_i(T) a_{iN} \right\}. \tag{2.35}$$

Пряме обчислення достовірності (2.32) призводить до втрати важливих розрядів, тому замість цього обраховують логарифм достовірності. При цьому замість рівняння (2.32) отримуємо:

$$\psi_j(t) = \max_i \left\{ \psi_i(t-1) + \log\left(a_{ij}\right) \right\} + \log\left(b_j(o_t)\right) \tag{2.36}$$

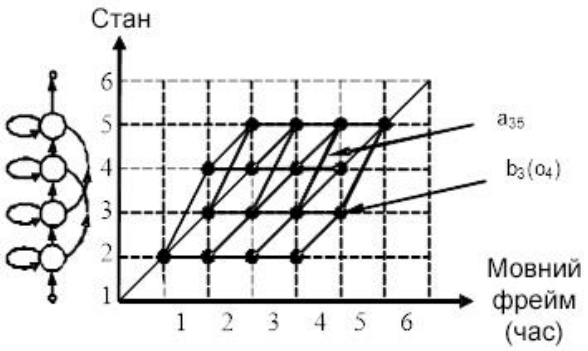


Рисунок 2.4. Алгоритм Вітербі для розпізнавання ізольованих слів

Це рекурентне співвідношення є базою для алгоритма Вітербі. На рис. 6 показано цей алгоритм, який можна представити як віднаходження оптимальнішого шляху через матрицю, де вертикальний вимір представляє стан ПММ, а горизонтальний вимір представляє часові проміжки мови. Велика точка на рисунку представляє логарифм ймовірності дослідження даного часового проміжку в даний момент часу, а кожен відрізок між точками відповідає логарифму ймовірності переходу. Логарифм ймовірності кожного можливого шляху обчислюється простим додаванням логарифмів ймовірностей переходів і логарифмів вихідних ймовірностей вздовж даного шляху. Шляхи йдуть зліва направо, колонка за колонкою. В конкретний час t, кожний шлях $\psi_i(t\text{-}1)$ відомий для всіх станів i, тому вираз (2.36) можна використовувати для обчислень $\psi_j(t)$, продовжуючи шляхи на один такт часу.

2.2. Параметризація мовних сигналів

Для розпізнавання мови, базованого на ПММ, необхідно отримати набір векторів дослідження. В якості векторів дослідження повинні бути обрані такі параметри, по яких можна достовірно відрізнити один звук мови від іншого.

Форма мовного сигналу одного і того ж звуку може суттєво змінюватися, тому відліки мовленнєвого сигналу безпосередньо не використовуються. Частіше використовують різні методи спектрального аналізу, засновані на перетворенні Фур'є, або на основі лінійного передбачення [12].

2.2.1. Аналіз на основі лінійного передбачення

Лінійне передбачення (LP) є одним з найефективних методів параметризації мовних сигналів. Даний метод домінує при оцінюванні основних параметрів мовленнєвого сигналу, таких як, період основного тону, форманта, спектр, функція площі мовного тракту. Домінантність методу обумовлена високою точністю отриманих оцінок і простою обчислень. Головний принцип методу лінійного передбачення полягає в тому, що поточний відлік мовного сигналу можна апроксимувати лінійною комбінацією попередніх відліків

$$s(n) = -\sum_{i=1}^{N}\left[a_i + s(n-i)\right] + e(n), \qquad (2.37)$$

де N – кількість коефіцієнтів моделі, $e(n)$ — помилка передбачення.

Коефіцієнти передбачення при цьому визначаються однозначно мінімізацією середнього квадрату $e(n)$ — різниці між відліками мовного сигналу та їх передбаченими значеннями.

Головні принципи методу лінійного передбачення добре узгоджуються з моделлю творення мови, за допомогою якого мовленнєвий сигнал можна представити у вигляді сигналу на виході лінійної системи зі змінними в часі параметрами, збуджувана квазіперіодичними імпульсами (в межах вокалізованого сегменту) або випадковим шумом (на невокалізованому сегменті). Метод лінійного передбачення гарантує точно і надійно оцінити параметри такої лінійної системи зі змінними коефіцієнтами. Передавальна функція такої системи:

$$H(z) = \frac{1}{\sum_{i=0}^{p} a_i z^{-i}} \tag{2.38}$$

де p - число полюсів і $a_0 \equiv 1$. Стосовно мовних сигналів існують наступні методи обчислення параметрів a_i (часто рівноцінні): коваріаційний, автокореляційний, сходового фільтру, зворотньої фільтрації, оцінки спектру, максимальної правдоподібності і скалярного добутку. [12]

2.2.2. Мел-кепстральні коефіцієнти

Окрім коефіцієнтів лінійного передбачення часто для розпізнавання мови використовують коефіцієнти кепстра відрізка мовного сигналу. Кепстр представляє собою Фур'є перетворення від логарифма спектра сигналу.

Такий «спектр спектра» дозволяє отримати характеристики мовного сигналу, які мінімально залежать від конкретної реалізації вимовленого слова. [15]

Назва "кепстр" ("cepstrum") отримано шляхом реверсії перших чотирьох букв слова спектр ("spectrum"). Вперше був визначений в 1963 Богертом (Bogert) та іншими. На відміну від класичного визначення кепстра, в літературі по обробці мови кепстр визначається як обернене перетворення Фур'є від логарифму спектра потужності сигналу. В математичному вигляді:

$$C_s = \mathrm{F}^{-1}\left\{\lg\left|\mathrm{F}\{s\}\right|\right\}, \tag{2.39}$$

де F – перетворення Фур'є, F^{-1} – обернене перетворення Фур'є, c – кепстр, s – початковий сигнал. На відміну від комплексного кепстра, визначений в **Ошибка! Источник ссылки не найден.** кепстр не містить інформацію про фазу сигналу. Змінна, від якої залежить кепстр, має розмірність часу, однак, це не той самий час, що у сигнала. Щоб підкреслити, що ця змінна в своєму роді частота для кепстра, іноді вживають назву quefrency (отримано від frequency). [15]

Операція обчислення кепстра відноситься до класу гомоморфної обробки сигналу. Гомоморфні системи – це клас нелінійних систем, які підкорюються загальному принципу суперпозиції. Лінійні системи – це частковий випадок гомоморфної системи. В обробці мови гомоморфні системи мають наступну властивість:

$$D\left[\left[x_1(n)\right]^{\alpha}\cdot\left[x_2(n)\right]^{\beta}\right]=\alpha D\left[x_1(n)\right]+\beta\left[x_2(n)\right]. \qquad (2.40)$$

Цей тип суперпозиції відноситься до операцій множення і піднесення до степеню. Такою узагальненою властивістю суперпозиції володіє функція логарифму.

Гомоморфні системи корисні в обробці мови, тому що вони надають метод для розділення форми збуджуючого сигналу і імпульсній перехідній характеристиці голосового тракту. Для розпізнання мови цей підхід цікавий з точки зору моделювання характеристик голосового тракту. Розділення двох компонент можна представити як процес деконволюції (обернена згортка), і може бути описаний наступним чином:

$$s(n)=g(n)\otimes v(n), \qquad (2.41)$$

де $g(n)$ – збуджуючий сигнал, $v(n)$ – імпульсна перехідна характеристика голосового тракту, а $\otimes$ - операція згортки. В частотному вигляді згортка представляється як:

$$S(f)=G(f)\cdot V(f). \qquad (2.42)$$

Якщо провести операцію комплексного логарифмування над обома частинами, то отримаємо:

$$\mathrm{Log}\left(S(f)\right)=\mathrm{Log}\left(G(f)\cdot V(f)\right)=\mathrm{Log}\left(G(f)\right)+\mathrm{Log}\left(V(f)\right). \qquad (2.43)$$

З цього моменту, в логарифмічному сенсі, збудження і характеристика голосового тракту представляють собою адитивну суміш. Обернене перетворення Фур'є цієї суміші і буде шуканий кепстр сигналу. Кепстр буде представляти собою суміш імпульсної характеристики і сигналу збудження, які при

необхідності можна розділити методами лінійної фільтрації. Інформація про голосовий тракт буде зосереджена, в більшості, в області малих часів кепстра, в той час як в області великих часів полягає інформація про сигнал збудження.

Замість обчислення Фур'є перетворення сигналу на практиці частіше за все користуються швидким перетворенням Фур'є або використовують гребінку фільтрів. Окрім того, кепстр сигналу також можна отримати з коефіцієнтів лінійного передбачення. Для цього використовується проста рекурсивна формула:

$$c_n = -a_n - \frac{1}{n}\sum_{i=1}^{n-1}(n-i)a_i c_{n-i} \tag{2.44}$$

Тут порядок кепстра не обов'язково рівний кількості коефіцієнтів LPC.

Відомо, що людське вухо має різну частотну роздільну здатність на різному діапазоні частот, іншими словами, висота звуку, яка сприймається людським слухом залежить від його частоти нелінійним чином. Існує думка, що таке нелінійне перетворення підвищує розбірливість мови. В системах розпізнавання мови для отримання аналогічного перетворення використовують гребінку фільтрів різної ширини. Одна з таких нелінійних шкал, апроксимуюча шкалу частот людського слуху, називається Мел-частотною. Вона визначається як:

$$\mathrm{Mel}(f) = 1127\ln\left(1+\frac{f}{700}\right). \tag{2.45}$$

Приведемо алгоритм отримання мел-кепстральних частотних коефіцієнтів:

1. Вихідний мовний сигнал запишемо в дискретному вигляді

$$x[n],\ 0 \le n < N. \tag{2.46}$$

2. Застосуємо до мовного сигналу перетворення Фур'є

$$X_a[k] = \sum_{n=0}^{N-1} x[n] e^{\frac{-2\pi i}{N}kn},\ 0 \le k < N. \tag{2.47}$$

3. Складемо гребінку трикутних фільтрів

$$H_m = \begin{cases} 0, & k < f[m-1]; \\ \dfrac{(k - f[m-1])}{(f[m] - f[m-1])}, & f[m-1] \le k < f[m]; \\ \dfrac{(f[m+1] - k)}{(f[m+1] - f[m])}, & f[m] \le k \le f[m+1]; \\ 0, & k > f[m+1]. \end{cases} \quad (2.48)$$

Приклад отриманої гребінки фільтрів для випадку $M = 12$ показаний на рис 7. [12]

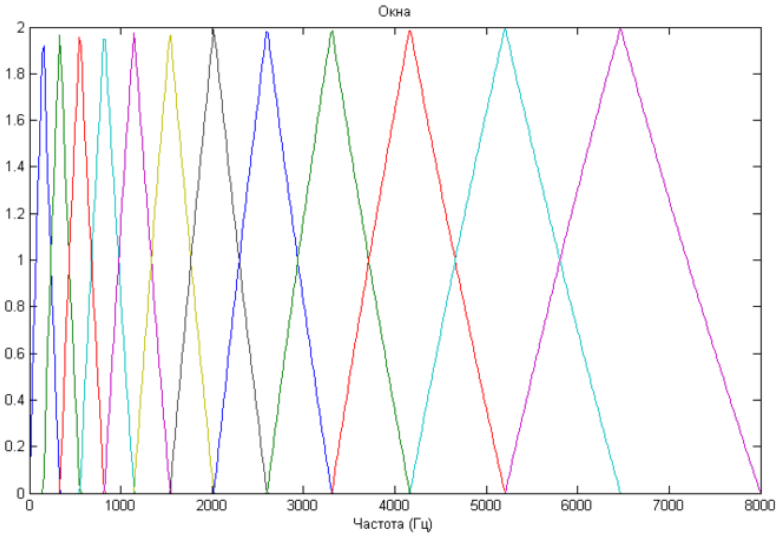


Рисунок 2.5. Гребінка фільтрів мел-частотної шкали ($M = 12$)

4. Для якої частоти $f[m]$ отримаємо з рівності

$$f[m] = \left(\frac{N}{F_s}\right) B^{-1}\left(B(f_1) + m\frac{B(f_h) - B(f_1)}{M+1}\right). \quad (2.49)$$

5. $B(b)$ — перетворення значення частоти в мел-шкалу, відповідно,

$$B^{-1}(b) = 700\left(e^{b/1125} - 1\right). \quad (2.50)$$

6. Обчислимо енергію для кожного вікна

$$S[m] = \ln\left(\sum_{k=0}^{N-1} \left|X_a[k]^2\right| H_m[k]\right), \ 0 \le m < M. \quad (2.51)$$

7. Застосуємо дискретне косинусне перетворення (перетворення Фур'є немає

сенсу використовувати, так як обчислення відбуваються над дійсними числами)

$$c[n] = \sum_{m=0}^{M-1} S[m] \cos\left(\pi n \frac{m + 0.5}{M} \right), \ 0 \le n < M. \quad (2.52)$$

Значення, отримані за допомогою виразу (2.52), є шуканими мел-кепстральними коефіцієнтами. [15]

Зазвичай, трикутні фільтри покривають весь частотний діапазон від нуля впритул до частоти Найквіста. Однак, обмеження полоси пропускання часто корисне, щоб відкинути небажані частоти або уникнути встановки границь фільтрів в частотних діапазонах, в яких немає корисної енергії сигналу [14].

2.2.3. Дельта-коефіцієнти

Ефективність системи розпізнавання мови може бути суттєво збільшена, якщо додатково використовувати похідну по часу від основних статичних параметрів. Ці коефіцієнти названі дельта коефіцієнтами першого порядку, другого порядку (прискорення) і третього порядку. Обчислюють коефіцієнти дельта за допомогою наступної формули:

$$d_t = \frac{\sum_{\theta=1}^{\Theta} \theta (c_{t+\theta} - c_{t-\theta})}{2 \sum_{\theta=1}^{\Theta} \theta^2}. \quad (2.53)$$

Коефіцієнти другого і третього порядку обчислюють по тій самій формулі, але вже від дельта коефіцієнтів попереднього порядку [12].

Висновки до розділу 2

Розглянуті математичні підходи до вирішення задачі розпізнавання, якими користувалися впродовж розвитку науки розпізнавання. Описано, як від опису моделі випадкового процесу ланцюгом Маркова, коли система в дискретні проміжки часу стрибково змінює свій стан, перейшли до прихованих марковських моделей, які являють собою статистичну модель, яка імітує неперервну роботу процесу з невідомими параметрами з задачею пошуку цих невідомих на основі спростерігаємих.

Роль системи розпізнавання полягає в тому, що вона має поставити в співвідношенні одна одній послідовності векторів параметрів мови і потрібні послідовності символів. Зробити це дуже важко по двом причинам. По-перше, перехід від символів до мови не є однозначним: різні символи можуть приводити до появи майже однакових звуків мови. Окрім того, значні зміни мовного сигналу можуть відбутися при зміні диктора, інтонації, обстановки, і т.д. По-друге, неможливо абсолютно точно вказати на мовному сигналі границі між символами. Тому мовний сигнал неможливо трактувати як послідовність з'єднаних незмінних образів.

Описані методи визначення параметрів прихованих марковських моделей за допомогою рекурентного оцінювання Баума-Уелча. Параметри ПММ рекурентно оцінюються по єдиній послідовності спостереження, тобто по єдиному екземпляру вимовленого слова. На практиці ж, для отримання хороших оцінок параметра, необхідно багато екземплярів одного і того ж слова. Тим не менш, використання багаторазових послідовностей спостереження не призводить до ускладнення алгоритму: процес визначення параметрів повторюється для кожної нової навчальної послідовності.

Розглянутий розпізнавання Вітербі та мел-кепстральні коефіцієнти.

РОЗДІЛ 3.
ПРОГРАМНІ ІНСТРУМЕНТИ ДЛЯ РЕАЛІЗАЦІЇ СИСТЕМИ РОЗПІЗНАВАННЯ МОВИ

3.1. Опис SaaS-системи розпінавання мови

В якості системи розпізнавання мови було обрано стратегію побудови клієнт-серверного програмного забезпечення (SaaS-системи) для автоматизації первинного огляду під час прийому пацієнтів лікарями. Метою побудованої системи є суттєве скорочення робочого часу медичних працівників, який ті, в свою чергу, витрачають на ручне заповнення інформації після первинного огляду пацієнта. В супротив такому підходу постала система розпізнавання мови, яка перетворює слова лікаря в текст і зберігає на сервері в особистому кабінеті пацієнта. Таким чином, замість годинного заповнення інформації, лікарям необхідно одразу ж на прийомі проговорювати системі необхідні дані для запису, таким чином суттєво заощаджувати робочий час.

3.2. Програмні інструменти для сховища даних

В якості сховища даних для автоматизованої системи було обрано реляційну базу даних PostgreSQL. Дану базу даних було обрано з переконань використання новітніх підходів щодо підвищення працездатності, відкритого коду, надійності та швидкості роботи. Також до переваг наведеної системи керування базами даних є наявність індексів, які пришвидшують пошук завдяки вбудованому алгоритму пошуку: Binary-Search-Tree. Іншою помітною перевагою бази даних є її можливість підтримки механізму Multiversion Concurrency Control (MVCC), завдяки яким база даних повністю відповідає вимогам транзакційності – ACID і

забезпечує зручний механізм, який, в більшості випадків не потребує блокування зчитування даних, що значно пришвидшує запити на вибірки даних одночасно кількома паралельними потоками.

3.3. Програмні інструменти для розробки серверної частини

В якості програмного інструмента для побудови веб-серверу був обраний популярний на даний час фреймворк Spring Boot, який являє собою програмну основу з відкритим кодом та контейнерами, які забезпечують підтримку інверсії управління для мови програмування Java. Обраний інструмент побудови веб-серверу забезпечує високу ефективність паралельної обробки запитів, швидкість розробки, переваги автоконфігурації основних компонентів та зручна побудова REST API для взаємодії з клієнтом.

В якості мови програмування була обрана Java, як сучасна мова програмування, яка є кросплатформною, що дозволяє її легкому переносу та розгортанню. Мова програмування Java, розгортаючись в Java Virtual Mashine дозволяє свою легку переносимість, підтверджуючи аспект кросплатформності. Для з’єднання веб-серверу з системою керування базами даних був обраний фреймворк Spring Data, який інкапсулює в собі потудні функціональні можливості Hibernate – ORM-фреймворку.

3.4. Програмні інструменти для розгортання SaaS-системи

Для розгортання розробленого прогрмного забезпечення був обраний популярний засіб для контейнеризаціїї – Docker, який забезпечує легкість розповсюдження програмного забезпечення, розгортання та підтримки.

3.5. Програмні інструменти для розробки клієнтської частини

Для розробки клієнтської частини програмного забезпечення був обраний фреймворк React JS. На даний момент він являється одним з найпопулярніших засобів для розробки клієнтської частини, за рахунок своєї швидкості роботи та побудови компонентів, а також завдяки особливій архітектурі функціонування.

Висновки до розділу 3

Розглянуто та описано систему для проектування, актуальність її побудови та застосування в реальному житті. Описані основні структурні частини, з яких вона побудована, а також програмні інструменти, які були застосовані для її реалізації. З описаного інструментарію можна зробити висновок, що система надійна, побудована з застосуванням засобів, пристосованих до сучасного навантаження, наприклад, великої кількості потоків водночас, різкій зміні середовища функціонування, апаратних помилок та суміжних аномалій.

РОЗДІЛ 4.
ПРАКТИЧНА РЕАЛІЗАЦІЯ ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ АВТОМАТИЧНОГО РОЗПІЗНАВАННЯ МОВИ

Зазвичай, при використанні системи розпізнавання мови мовленнєві сигнали сприймаються за допомогою мікрофонів побутового рівня, в несприятливих для запису мови умовах. Одночасно, комерційні системи розпізнавання мови, не передбачаючи додаткового навчання безпосередньо користувачем, використовують в основі свого мовного корпусу записи, записані, зазвичай, на високоякісній апаратурі і у відповідних умовах. Внаслідок розбіжностей технічних і природній каналів відбувається погіршення якості розпізнавання мови.

Найпростіший шлях запису, який достатньо використовується в побуті користувачами персональних комп'ютерів, представлений на рис. 4.1

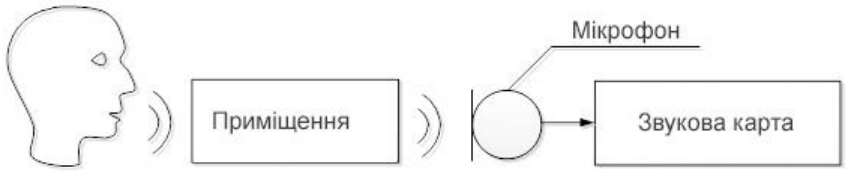


Рисунок 4.1. Тракт запису мовного сигналу

4.1. Нормалізація АЧХ вхідного тракту

На сьогоднішній день, задача компенсації відмінностей між властивостями та характеристиками різних мікрофонів, досліджена доволі погано. Також, фахівцями в області обробки кольорових зображень аналогічна задача була вирішена застосуванням «кольорових профілів» приладів (таких як фотокамери,

принтери, сканери, монітори та інші). «Кольоровий профіль» для сканера — це калібрувальна таблиця, яка дозволяє отримати відомості про різницю між кольоровим простором даного сканера і деяким іншим кольоровим простором-еталоном. Використовуючи ці дані, можна програмним шляхом компенсувати нерівності в кольоровій чутливості сканера. Також, відомі методи створення «кольорових профілів» часто потребують наявності спеціальних умов освітлення і використання коштовного колориметра, або ж надзвичайно часозатратні і не гарантують коректності отриманих результатів. [16]

Найближчим аналогом сканера в області апаратури для роботи зі звуком є мікрофон. Провівши аналогії з процесом створення «кольорових профілів», в даному підрозділі запропонований метод отримання «звукового профіля» мікрофона. Метод засновується на вивчені АЧХ мікрофона, для якого складається «звуковий профіль». Внаслідок, для його створення необхідно визначити АЧХ мікрофона якомога більш точним чином. Найбільш підходящими методами в контексті цілі дослідження представляють метод безпосереднього виміру і метод порівняння.

4.1.1. Метод безпосереднього виміру

При отриманні АЧХ мікрофона даний метод передбачає що запис відбувається в умовах вільного поля (наприклад, в безеховій камері), а АЧХ випромінювача рівномірна по всьому діапазону частот, які здатна сприйняти людина. Сигнал, що випромінюється, являє собою представлення білого шуму тривалістю не менше 30 с. Рівень вихідного сигналу випромінювача встановлюється таким чином, щоб на виході мікрофону було відсутнє перевантаження за рівнем. Досліджуваний мікрофон встановлюється на відстані 1 м від випромінювача так, щоб акустичні вісі випромінювача і мікрофона були

перпендикулярні одне одному.

Безпосередньо АЧХ мікрофона можна отримати, взявши модуль перетворення Фур'є записаного шуму

$$A_{мик} = \left| H_{мик}(f) \right| = \left| \mathrm{F}\{Y(t)\} \right|, \quad (4.1)$$

де F — дискретне перетворення Фур'є, $Y(t)$ — сигнал на виході мікрофону.

Даний метод є найточнішим методом отримання АЧХ та має використовуватись завжди, якщо є можливість забезпечити необхідні умови запису. Для запису в домашніх умовах допустимим вважається використовувати приміщення з невеликою кількістю відзеркалюючих поверхонь, а в якості випромінювача для відтворення білого шуму — якісну широкополосну акустичну систему.

4.1.2. Метод порівняння

Метод порівняння є відносним методом оцінки АЧХ мікрофона. В реальних експериментах за допомогою даного методу можна отримати добрі результати, хоч і не такі точні, порівнюючи їх з методом безпосереднього виміру.

Суть методу полягає в використанні додаткового еталонного мікрофона з завчасно відомою АЧХ.

При використанні методу порівняння не висуваються настільки ж суворі вимоги до умов запису і використаній апаратурі. Використовуючи інший мікрофон, простим відніманням спектрів можна з'ясувати, яких змін зазнає спектр в порівнянні з еталонним для цього мікрофону АЧХ, отриманим виробником методом безпосереднього виміру, або будь-яким іншим. Таким чином, вираз для АЧХ досліджуваного мікрофона можна записати як

$$A_{мик} = \left| H_{мик}(f) \right| = \left| \mathrm{F}\{Y(t)\} \right| + \left(A_{эт.АЧХ} - \left| \mathrm{F}\{Z(t)\} \right| \right), \quad (4.2)$$

де $Y(t)$ — сигнал на виході досліджуваного мікрофона; $A_{эт.АЧХ}$ — апріорно відома АЧХ еталонного мікрофона; $Z(t)$ — сигнал на виході еталонного мікрофона.

4.1.3. Отримання звукового профіля мікрофону

Незалежно від методу отримання АЧХ мікрофону, вираз для розрахунку «звукового профілю» досліджуваного мікрофона можна записати як

$$K_{зв.проф.} = \frac{A_{бел.шум}}{A_{мик}}, \tag{4.3}$$

де $A_{бел.шум}$ — амплітудний спектр білого шуму. Однак, так спектр білого шуму в ідеалі представляє собою константу, вираз **Ошибка! Источник ссылки не найден.** можна записати як

$$K_{зв.проф.} = \frac{1}{A_{мик0}}, \tag{4.4}$$

де $A_{мик0}$ — приведена до одиниці АЧХ мікрофона, яку можна отримати за допомогою виразу

$$A_{мик0} = \frac{A_{мик}}{\max(A_{мик})}, \tag{4.5}$$

де $\max(A_{мик})$ — амплітудне значення АЧХ мікрофона.

Приклад того, як може виглядати такий «звуковий профіль», приведений на рис. 4.2. В якості досліджуваного мікрофона використовувався мікрофон мобільного телефону Motorola Defy.

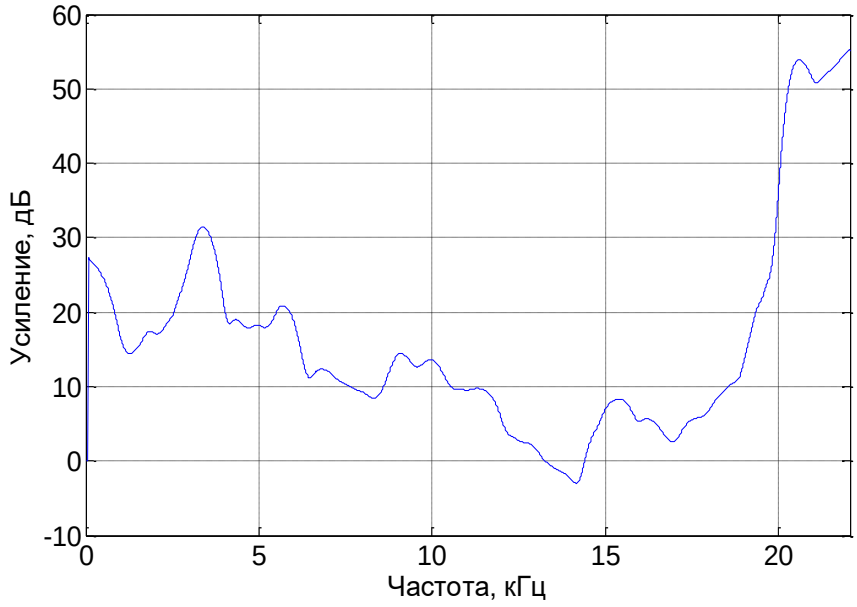


Рисунок 4.2. Приклад «звукового профілю»

Як видно з рис. 2. Рисунок , даний мікрофон демонструє значну нерівномірність АЧХ на всьому спектрі частот.

Спосіб застосування «звукового профіля» до записаного сигналу описується наступним виразом:

$$N_{корр}(t) = \mathrm{F}^{-1}\left\{\mathrm{F}\left\{X(t)\right\}\cdot K_{зв.проф}\right\}. \quad (4.6)$$

де $N_{корр}(t)$— сигнал, до якого застосований «звуковий профіль»; $X(t)$ — сигнал, підвержений корекції.

4.2. Усунення клацань і піків в сигналі

Наявність в мовному сигналі коротких «клацань» і звукових «сплесків», що виникають при відкритті рота, призводить до значного підвищення кількості помилкових спрацьовувань системи.

Був розроблений фільтр, дозволяючий ефективно усувати подібні артефакти в сигналі. Принцип роботи фільтра можна описати наступним алгоритмом.

1. Позиція початку фрейму $t_н$ встановлюється в нульовий момент часу.

2. З сигналу виділяється фрейм $F_{щел}$, який ймовірно містить «клацання». Розраховується його енергія $E_{щел}$.

$$F_{щел} = [t_н; t_н + \tau_ф], \tag{4.7}$$

де $\tau_ф$ — передбачувана тривалість клацання.

Вираз для розрахунку енергії відрізка дискретного сигналу має вигляд

$$E = \sum_{n_1}^{n_2} x_i^2, \tag{4.8}$$

де n_1, n_2 — номери відліків початку і кінця фрейма, x_i — i-тий відлік сигналу.

3. Виділяються фрейми зліва і справа від фрейма $F_{щел}$. Позначимо їх $F_{лев}$ і $F_{прав}$ відповідно.

$$\begin{aligned} F_{лев} &= [t_н - \tau_{ан}; t_н]; \\ F_{прав} &= [t_н + \tau_ф; t_н + \tau_ф + \tau_{ан}], \end{aligned} \tag{4.9}$$

де $\tau_{ан}$ — тривалість фрейма аналізу.

Прийняття позитивного рішення про наявність клацання в фреймі $F_{щел}$ приймається у випадку істинності рівності

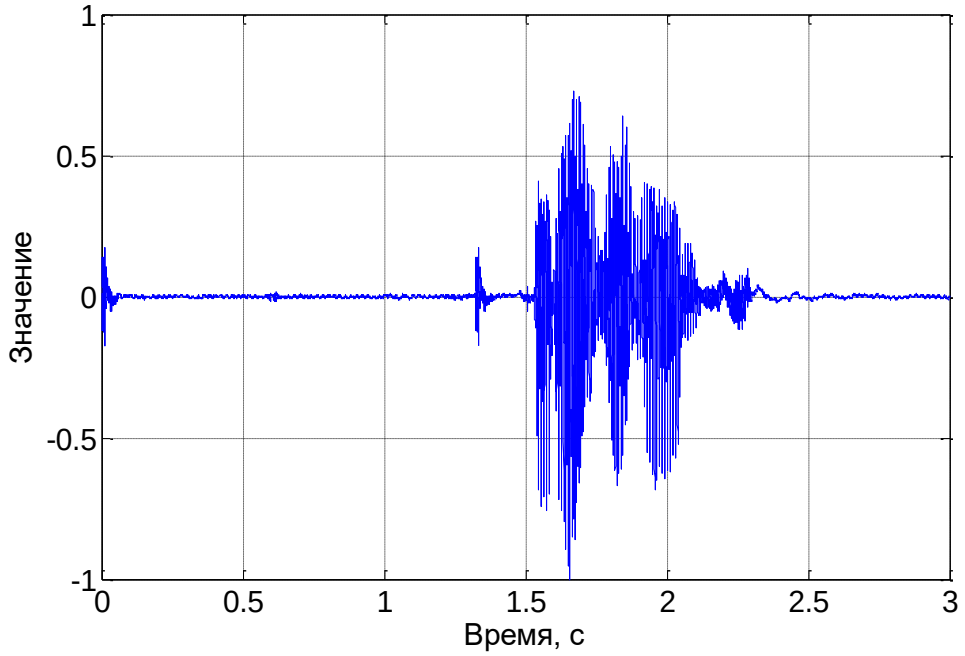
$$E_{щел} > (E_{лев} + E_{прав}) \cdot k, \tag{4.10}$$

де k – деякий коефіцієнт, який залежить від $\tau_ф$ і $\tau_{ан}$.

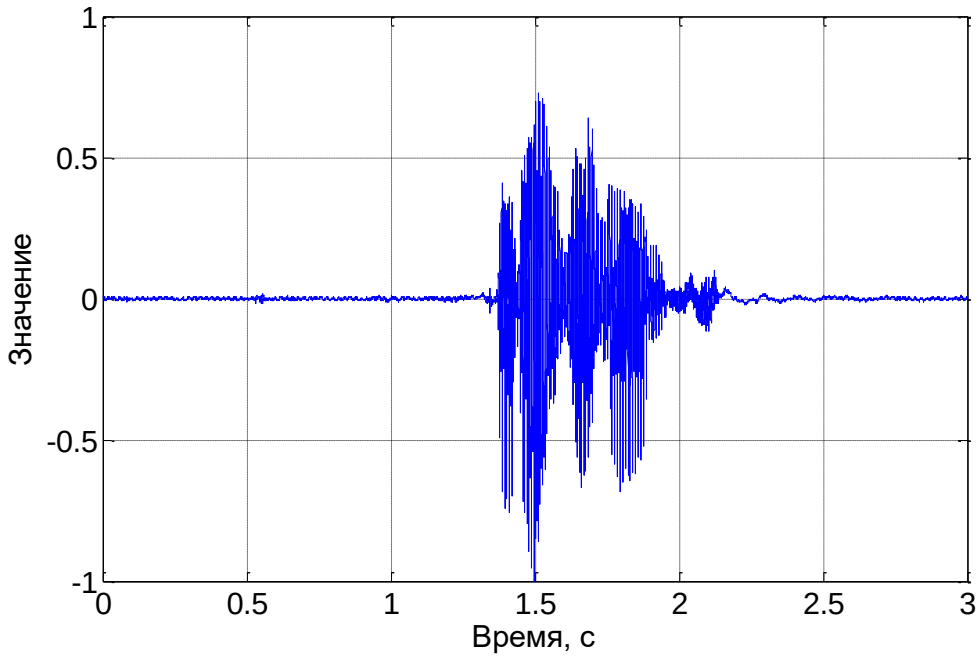
4. Позиція початку фрейма $t_н$ зсувається вперед на час $\tau_н$.

5. Повторяти обробку з п. 2 до тих пір, поки не буде досягнутий кінець сигналу.

Приклад результатів обробки приведений на рис. 4.3.



а)



б)

Рисунок 4.3. Результат роботи детектора «клацань»

В даному випадку виявились усунуті два сплески: на початку і в кінці запису, в той час як інша частина сигналу залишилась неторкнутою.

Висновки до розділу 4

В розділі розглянуті реальні проблеми, які зустрічаються при записуванні голосового сигналу. А несприятливі умови для запису, які характеризуються наявністю сторонніх шумів. В якості дослідження можливих варіантів вирішення, або, принаймні, зниження рівня пошкоджень було розглянуто такі теми, як нормалізація вхідного сигналу та виділення основного сигналу, а також методи щодо усунення деяких яскраво видимих перешкод в записаному голосовому сигналі.

ВИСНОВКИ

В дипломному проекті розглянуто актуальність обраної теми, її розвиток впродовж минулого та поточного століття, виявлення різноманітних перешкод та тенденція способів їх вирішення. Таким чином, описано перші спроби розумного застосування розпізнавання голосового сигналу для управління технікою, що дозволяє керувати в приданому стилі та заощаджує час. Показані можливі небезпечні фактори, які можуть виникати в разі широкого застосування такого способу управління, разом з тим виявлені проблеми, які стоять перед науковцями, вирішення яких дозволить частково, а то й повністю уникнути таких факторів.

Розглянуті різні методи, якими користувались в минулому та які удосконалюються на набувають нових масштабів в сучасній науці по розпізнаванню голосового сигналу.

В дипломному проекті побудована автоматизована система, яка все більше й більше набуває актуальності в сучасності. Дана система являє собою SaaS-систему для медичних установ, а саме ефективний допоміжний програмний засіб для медичних працівників. Наразі, система налаштована на автоматизацію занесення інформації про первинний огляд пацієнтів до їх електронної медичної книжки, проте, система підлягає масштабизації та розширенню, для прикладу, вона може бути автоматизована також для подальших оглядів та постійного її використання для проведення лікування. Втілення такої функціональності потребує покращення розпізнавання голосового сигналу в зашумленому середовищі, яким є зазвичай воно в реальності, а також додавання можливості налаштування системи на конкретний голос в певний проміжок часу. Вказані проблеми наразі є глобальними та підлягають дослідженню та пошуку алгоритмів їх вирішення.

ПЕРЕЛІК ПОСИЛАНЬ

1. Connecticut Product of the Day: Radio Rex! // Museum of Connecticut History URL: https://ctinventor.wordpress.com/2012/01/13/connecticut-product-ofthe-day-radio-rex/;

2. IBM Shoebox // IBM Archives URL: http://www03.ibm.com/ibm/history/ exhibits/specialprod1/specialprod1_7.html;

3. Интерактивное голосовое редактирование текста с помощью новых речевых технологий от Яндекса//URL: https://habrahabr.ru/

company/yandex/blog/243813);

4. Распознавание русской речи для колл-центров // URL: https://habrahabr.ru/company/croc/blog/235565/;

5. OneBox переводит телефонные разговоры в текст и проверяет каждое слово // nastol.ru URL: http://www.nastol.ru/Go/ViewNews?id=79754;

6. Яндекс.Разговор // GitHub URL: https://github.com/yandex/deaf;

7. Речевые технологии Speech Kit // Яндекс разработчикам URL:

8. Massimo Basile, Mario Fabrizi CMU Sphinx: the recognizer library. M.: Sapienza University di Roma, 2013. 13c.;

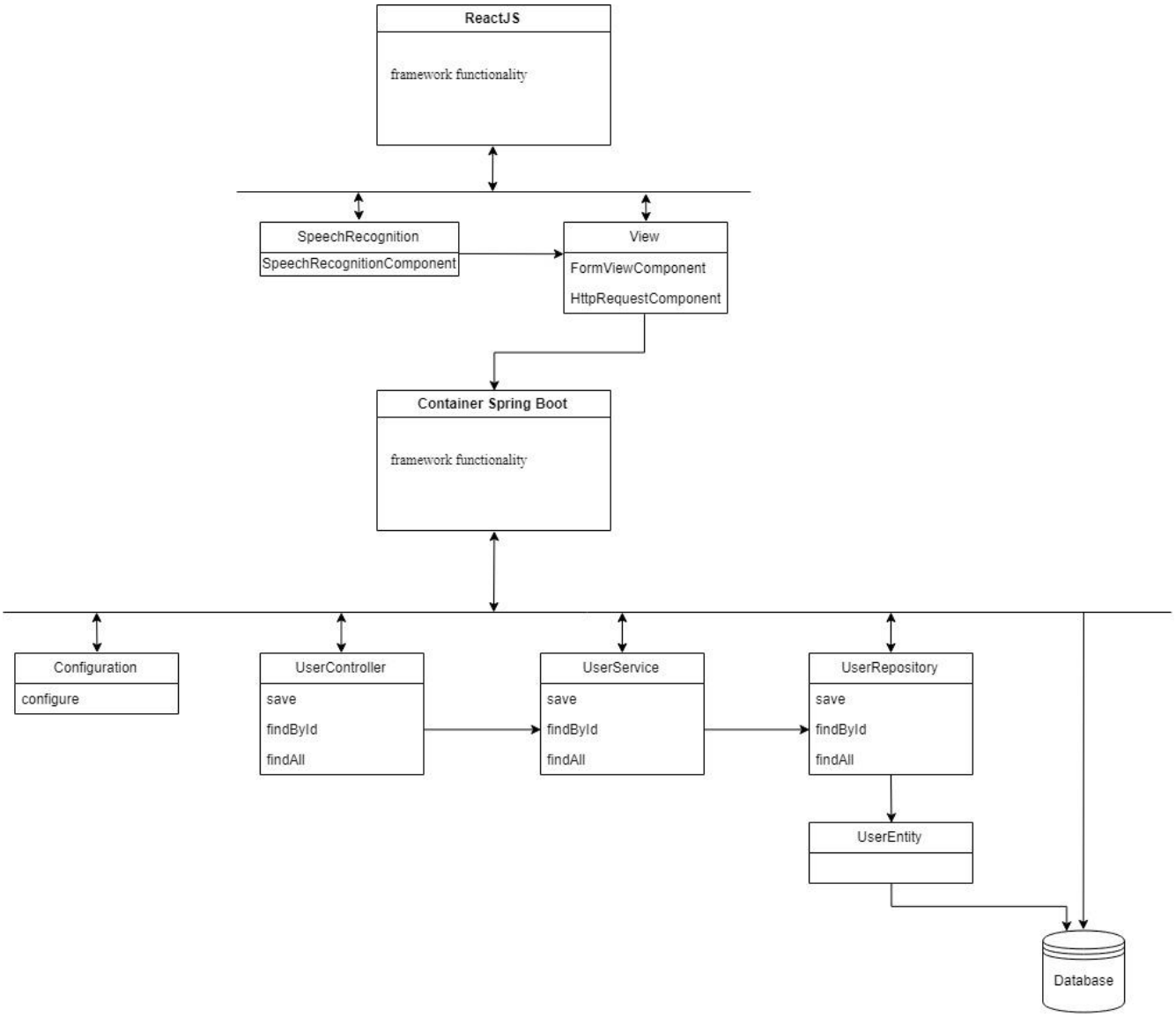
9. Julius //GitHub URL: https://github.com/julius-speech/julius

10. Математичні методи рішення воєнно-спеціальних задач / В. З. Казачинський, Г.Е. Левитський. — Москва, 1980.

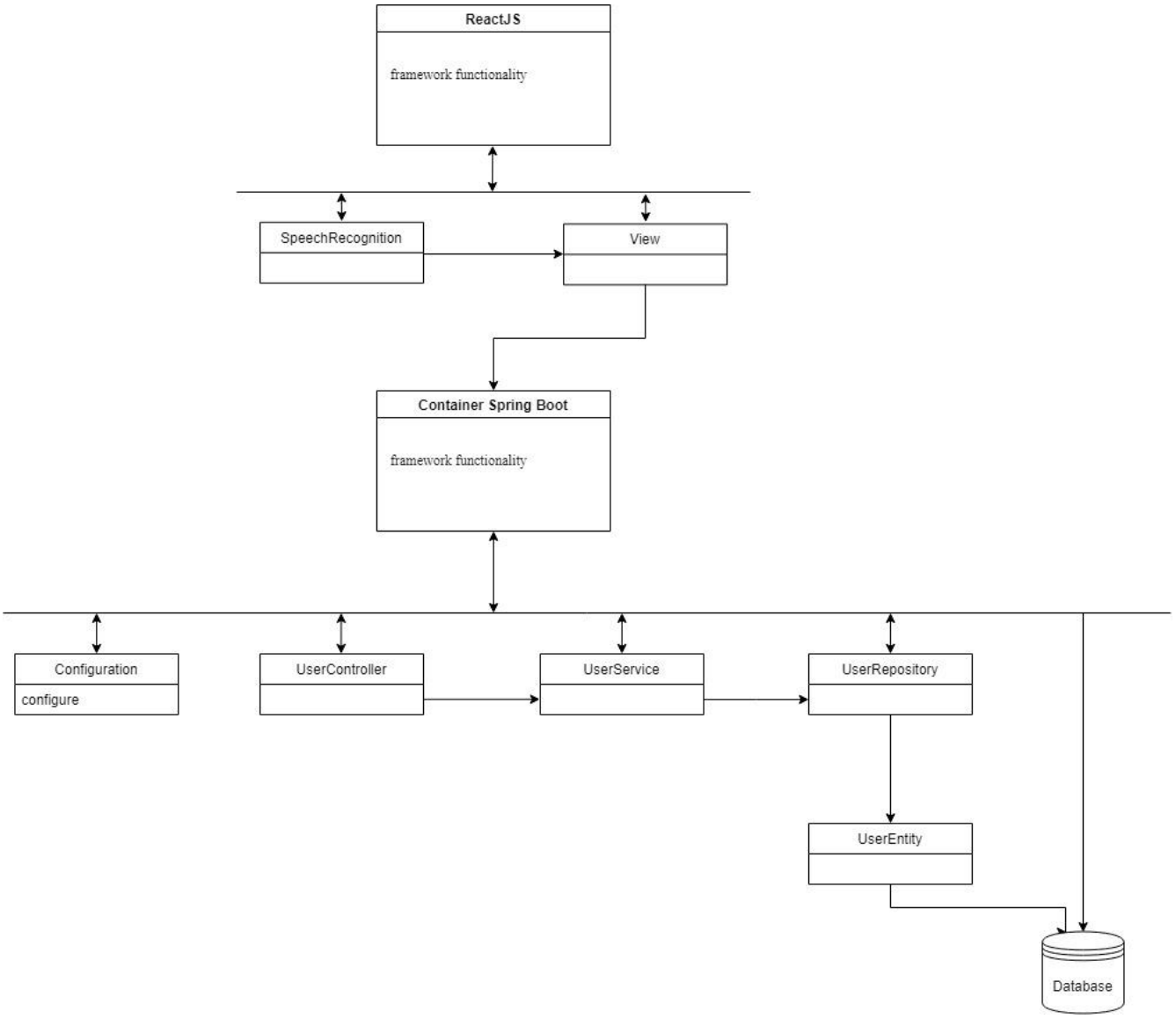
11. Прихована марковська модель — Вікіпедія. [Електронний ресурс], http:// uk.wikipedia.org/wiki/Прихована_марковська_модель

12. Дідковський В. С., Продеус А. М., "Дослідження шляхів підвищення ефективності комп'ютерних систем слухомовної корекції людей з порушеннями слуху та глухотою," НТУУ «КПІ», Київ, 2008.
13. Сичов А. В. Приховані марковські моделі — Вікіпідручник. [Електронний ресурс], http://ru.wikibooks.org/wiki/Скрытые_марковские_модели
14. "The HTK Book". [Електронний ресурс], http://htk.eng.cam.ac.uk/docs/docs.shtml
15. Урицький І. Хабрахабр. [Електронний ресурс], http://habrahabr.ru/post/140828/,
16. ICC profile — Wikipedia. [Електронний ресурс], http://en.wikipedia.org/wiki/ICC_profile
17. Охорона праці користувачів комп'ютерних відеодисплейних терміналів / Навакатикян А.О., Кальниш В.В. — Київ, Україна, 1997. — с. 400.
18. Основи охорони праці: Підручник. 2-ге видання, доповнене та перероблене. / К. Н. Ткачук, М. О. Халімовський, В. В. Зацарний, Д. В. Зеркалов, Р. В. Сабарно, О. І. Полукаров, В. С. Коз'яков, Л. О. Мітюк. За ред. К. Н. Ткачука і М. О. Халімовського. — КиевОснова, 2006. — с. 448.

Додатки



					ІАЛЦ 462619.004 Д1		
Зм.	Арк.	№ докум.	Підпис	Дата			
Розробив		Стефанішина У.С.			Діаграма класів	Лит.	Аркуш 1 / Аркушів 1
Перевірив		Сімоненко В.П.					
Т. Контр.						НТУУ “КПІ ім. Ігоря Сікорського” ФІОТ гр ІО-52	
Н. Контр.		Сімоненко В.П.					
Затвердив		Сімоненко В.П.					



					ІАЛЦ 462619.005 Д2	
Зм.	Арк.	№ докум.	Підпис	Дата		
Розробив		Стефанішина У.С.			Структурна діаграма	Лит. / Аркуш 1 / Аркушів 1
Перевірив		Сімоненко В.П.				
Т. Контр.						НТУУ "КПІ ім. Ігоря Сікорського" ФІОТ гр ІО-52
Н. Контр.		Сімоненко В.П.				
Затвердив		Сімоненко В.П.				

					ІАЛЦ 462619.006 ДЗ		
Зм.	Арк.	№ докум.	Підпис				
Розробив		Стефанішина У.С.			Алгоритм розпізнавання голосу	Лит.	Аркуш 1 / Аркушів 1
Перевірив		Сімоненко В.П.					
Т. Контр.						НТУУ “КПІ ім. Ігоря Сікорського” ФІОТ гр ІО-52	
Н. Контр.		Сімоненко В.П.					
Затвердив		Сімоненко В.П.					

ДОДАТОК Д
Код програми

```
package com.diploma.speech.recognition.config;
import com.fasterxml.jackson.databind.ObjectMapper;
import com.fasterxml.jackson.databind.PropertyNamingStrategy;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
@Configuration
public class CustomConfiguration {
    @Bean
    public ObjectMapper objectMapper() {
        ObjectMapper objectMapper = new ObjectMapper();
        objectMapper.setPropertyNamingStrategy(PropertyNamingStrategy.SNAKE_CASE);
        return objectMapper;
    }
}
package com.diploma.speech.recognition.controller;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
@Controller
public class WebPageController {

    @RequestMapping(value = "/")
    public String index() {
        return "index";
    }
}
```

```
package com.diploma.speech.recognition.entity;

import lombok.EqualsAndHashCode;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import javax.persistence.*;

@Getter
@Setter
@NoArgsConstructor
@EqualsAndHashCode(of = "id")
@Entity
@Table(name = "users")
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(name = "first_name")
    private String firstName;

    @Column(name = "last_name")
    private String lastName;

    private Long age;
```

```
package com.diploma.speech.recognition.repository;

import com.diploma.speech.recognition.entity.User;
import org.springframework.data.jpa.repository.JpaRepository;

public interface UserRepository extends JpaRepository<User, Long> {
}
package com.diploma.speech.recognition;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.ComponentScan;

@SpringBootApplication
public class SpeechRecognitionApplication {

    public static void main(String[] args) {
        SpringApplication.run(SpeechRecognitionApplication.class, args);
    }
}
import React from 'react';
import './App.css';
import Artyom from 'artyom.js';
import ArtyomCommandsManager from './ArtyomCommandsManager.js';

import SideNav, {Toggle, Nav, NavItem, NavIcon, NavText} from '@trendmicro/react-sidenav';

import '@trendmicro/react-sidenav/dist/react-sidenav.css';
```

```
const Jarvis = new Artyom();

export default class App extends React.Component {
    constructor(props, context) {
        super(props, context);

        this.startAssistant = this.startAssistant.bind(this);
        this.stopAssistant = this.stopAssistant.bind(this);
        this.speakText = this.speakText.bind(this);
        this.handleTextareaChange = this.handleTextareaChange.bind(this);

        // Prepare simple state
        this.state = {
            artyomActive: false,
            textareaValue: "",
            artyomIsReading: false
        };

        // Load some commands to Artyom using the commands manager
        let CommandsManager = new ArtyomCommandsManager(Jarvis);
        CommandsManager.loadCommands();
    }

    startAssistant() {
        let _this = this;

        console.log("Artyom succesfully started !");

        Jarvis.initialize({
            lang: "ru-RU",
```

```
            debug: true,
            continuous: true,
            soundex: true,
            listen: true
        }).then(() => {
            // Display loaded commands in the console
            console.log(Jarvis.getAvailableCommands());

            Jarvis.say("Добро пожаловать в медицынский кабинет. Начинайте вводить данные. Имя пациента.");

            _this.setState({
                artyomActive: true
            });
        }).catch((err) => {
            console.error("Oopsy daisy, this shouldn't happen !", err);
        });
    }

    stopAssistant() {
        let _this = this;

        Jarvis.say("Спасибо. До встречи!");
        Jarvis.fatality().then(() => {
            console.log("Jarvis has been succesfully stopped");

            _this.setState({
                artyomActive: false
            });
```

```
    }).catch((err) => {
        console.error("Oopsy daisy, this shouldn't happen neither!", err);

        _this.setState({
            artyomActive: false
        });
    });
}

speakText() {
    let _this = this;

    _this.setState({
        artyomIsReading: true
    });

    // Speak text with Artyom
    Jarvis.say(_this.state.textareaValue, {
        onEnd() {
            _this.setState({
                artyomIsReading: false
            });
        }
    });
}

handleTextareaChange(event) {
    this.setState({
        textareaValue: event.target.value
    });
```

```
}

render() {
    return (
        <div className="App">
            <header className="App-header">

                <SideNav
                    onSelect={() => {
                        console.log("JJJ");
                        // Add your code here
                    }}>
                    <SideNav.Toggle/>
                    <SideNav.Nav defaultSelected="home">
                        <NavItem eventKey="home">
                            <NavIcon>
                                <i className="fa fa-fw fa-home" style={{fontSize: '1.75em'}}/>
                            </NavIcon>
                            <NavText>
                                Добавление пациента
                            </NavText>
                        </NavItem>
                        <NavItem eventKey="charts">
                            <NavIcon>
                                <i className="fa fa-fw fa-line-chart" style={{fontSize: '1.75em'}}/>
                            </NavIcon>
                            <NavText>
                                Список всех пациентов
                            </NavText>
                        </NavItem>
```

```
                </SideNav.Nav>
            </SideNav>
            <h1>Медицынский кабинет</h1>
<input className="form-style-5" type="button" value="Начать" disabled={this.state.artyomActive}
                onClick={this.startAssistant}/>

            <textarea id="FirstName" className="textArea" rows="2" defaultValue="Имя"/> <br/>
            <textarea id="MiddleName" className="textArea" rows="2" defaultValue="Отчество"/> <br/>
            <textarea id="LastName" className="textArea" rows="2" defaultValue="Фамилия"/> <br/>
            <textarea id="Age" className="textArea" rows="2" defaultValue="Возраст"/> <br/>
            <textarea id="Diagnoz" className="textArea" rows="2" defaultValue="Диагноз"/> <br/>
            {/*<div className="textarea-container">*/}
            {/*    <textarea id="Diagnoz" rows="5" defaultValue="Диагноз"*/}
            {/*            onChange={this.handleTextareaChange} value={this.state.textareaValue}/>*/}
            {/*    <button disabled={this.state.artyomIsReading}
onClick={this.speakText}>Прочитать</button>*/}
            {/*</div>*/}
            <br/>
            <input className="form-style-5" type="button" value="Закончить"
disabled={!this.state.artyomActive}
                onClick={this.stopAssistant}/>
        </header>
      </div>
    )
  }
}
```

export default class ArtyomCommandsManager {

  constructor(ArtyomInstance) {

```
    this._artyom = ArtyomInstance;
}

loadCommands() {
    let Artyom = this._artyom;

    return Artyom.addCommands([
        {
            indexes: ["Имя *"],
            smart: true,
            action: (i, wildcard) => {
                console.log("WAS TOLD" + wildcard);
                Artyom.say(wildcard);
                Artyom.say("Спасибо. Произнесите отчество пациента");
                document.getElementById("FirstName").value = wildcard;
            }
        },
        {
            indexes: ["Отчество *"],
            smart: true,
            action: (i, wildcard) => {
                console.log("WAS TOLD" + wildcard);
                Artyom.say(wildcard);
                Artyom.say("Спасибо. Произнесите фамилию пациента");
                document.getElementById("MiddleName").value = wildcard;
            }
        },
        {
            indexes: ["Фамилия *"],
            smart: true,
```

```
    action: (i, wildcard) => {
        console.log("WAS TOLD" + wildcard);
        Artyom.say(wildcard);
        Artyom.say("Спасибо. Произнесите возраст пациента");
        document.getElementById("LastName").value = wildcard;
    }
},
{
    indexes: ["Возраст *"],
    smart: true,
    action: (i, wildcard) => {
        console.log("WAS TOLD" + wildcard);
        Artyom.say(wildcard);
        Artyom.say("Спасибо. Раскажите о диагнозе пациента");
        document.getElementById("Age").value = wildcard;
    }
},
{
    indexes: ["Диагноз *"],
    smart: true,
    action: (i, wildcard) => {
        console.log("WAS TOLD" + wildcard);
        Artyom.say(wildcard);
        Artyom.say("Спасибо. Подтвердите данные.");
        document.getElementById("Diagnoz").value = wildcard;
    }
},
{
    indexes: ["Отправить"],
    action: () => {
```

```
                Artyom.say("Спасибо. Данные сохранены. До свидания.");
            }
        },
        {
            indexes: ["Привет! Как дела ?"],
            action: () => {
                Artyom.say("Спасибо, что спросили. У меня все хорошо. Надеюсь, никто не болеет.");
            }
        }
    ]);
  }
}
```